iris UNIVERSE

winter 1989

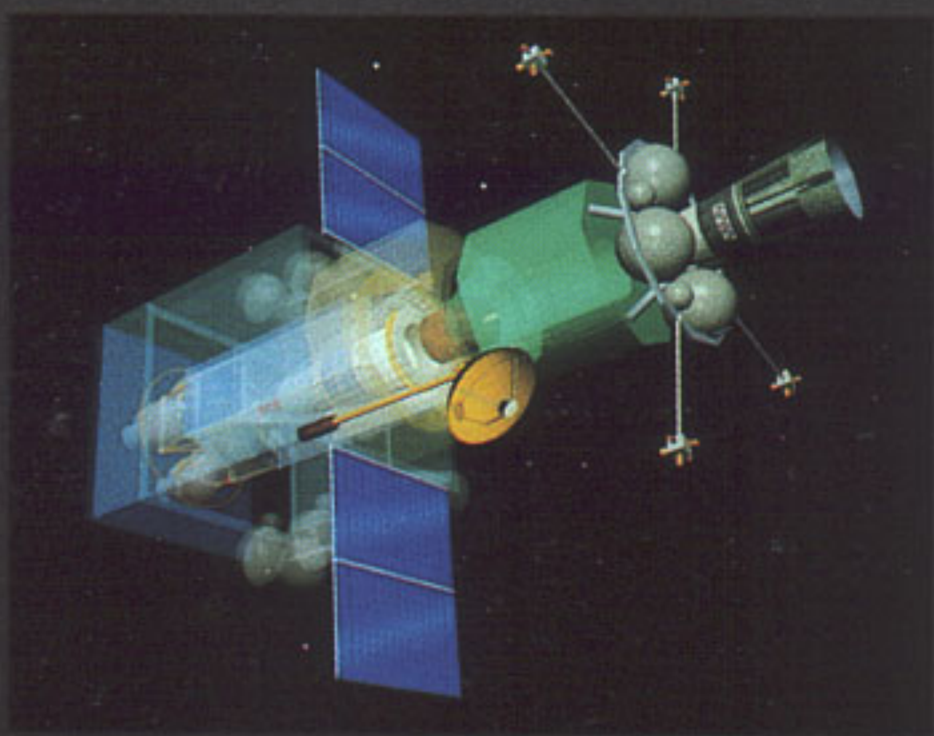
the IRIS community magazine

# With Clients Like These, Who Needs Copy?

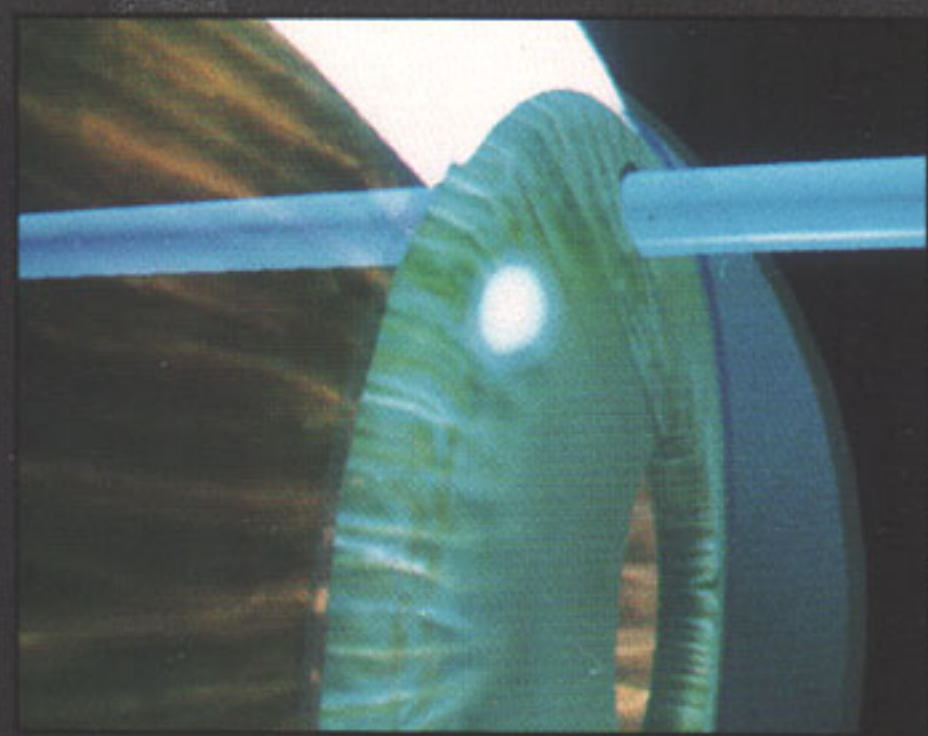
Partial Engineering/ Scientific Client List:

ADVANCED IMAGING
ATR COMMUNICATIONS SYSTEMS
AVERETT & ASSOCIATES
BARCO CREATIVE SYSTEMS
BIOGRAPHICS
BOEING ENGINEERING SCIENTIFIC
BOEING TECHNOLOGY CENTER
CHIYODA INSTITUTE OF TECHNOLOGY
COMPUTERVISION
CORNELL UNIVERSITY
ESTEC
FAILURE ANALYSIS ASSOCIATION
FORENSIC TECHNOLOGIES INC.
GENERAL DYNAMICS
HEWLETT PACKARD
HITECH LABS
INTEL CORPORATION
JET PROPULSION LABORATORY
MCDONNELL DOUGLAS
MORTON THIOKOL
NASA/DRYDEN
NASA/GODDARD
NASA/LBJ SPACE CENTER
NORTHROP RESEARCH AND TECHNOLOGY
NORTHROP AIRCRAFT DIVISION
NAVAL SURFACE WARFARE CENTER
NIPPON TELEGRAPH AND TELEPHONE
PRATT & WHITNEY
RTL PRODUCTIONS
SALT RIVER PROJECT
S.E.A.
SESA
SUMISHO ELECTRONIC SYSTEMS
TEXAS A&M UNIVERSITY
THE MITRE CORPORATION
TRW
UNIVERSITE LIBRE DE BRUXELLES
U.C.L.A.
UNIVERSITY OF ILLINOIS
UNIVERSITY OF MINNESOTA
UNIVERSITY OF TOKYO
U.S. AIR FORCE
VOLVO CAR CORPORATION

McDONNELL DOUGLAS MD11, DOUGLAS AIRCRAFT CORP. © 1988

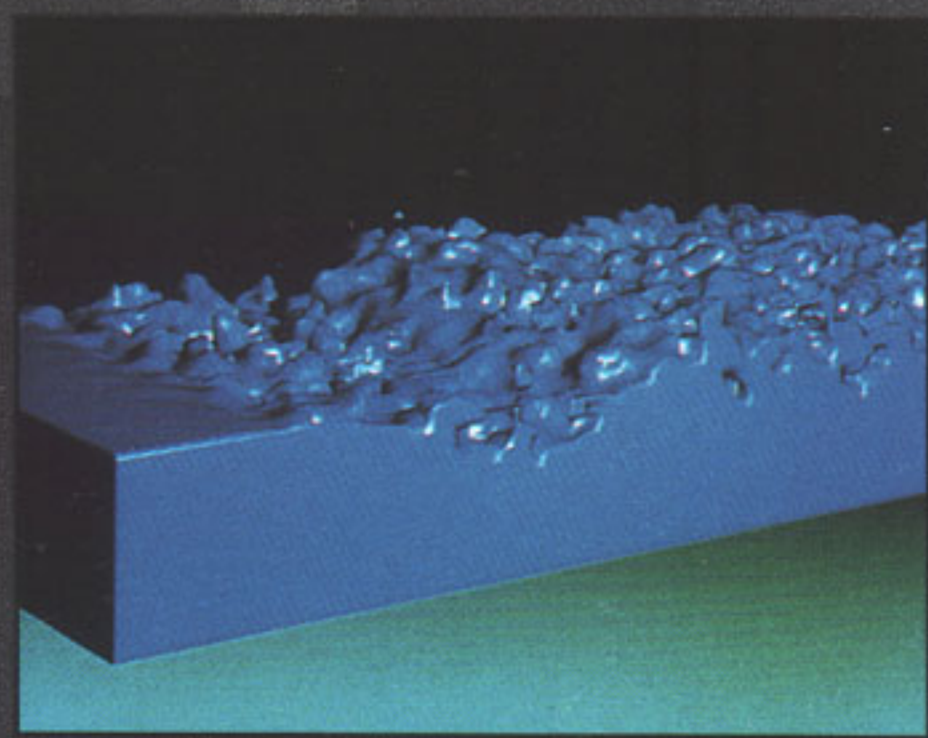


NASA/JSC/MPAD, "MARS SAMPLE RETURN MISSION"
INTERACTIVE GRAPHICS LAB DIRECTOR: GUNTER SABIONSKI
CHIEF ANIMATOR: ED KRAMER



MEDICAL RESEARCH: DAVID BOLINSKY
ADVANCED IMAGING © 1988

ROVER STERLING AUTOMOBILE
COMPUTERVISION, IGES DATABASE



"3D TURBULENT SHEAR FLOW AT MACH 1.1"
SCIENTIFIC RESEARCH: MICHAEL NORMAN, NCSA
VISUALIZATION: CRAIG UPSON, NCSA

## 3D Software For Animation And Simulation.

Images Developed Utilizing Silicon Graphics Super Workstation.

WAVEFRONT TECHNOLOGIES

*See* What You Think.

530 EAST MONTECITO STREET, SANTA BARBARA, CALIFORNIA 93103 805/962-8117

iris UNIVERSE


**editor: Mark Compton**
**production: Monica Schulze**

**design:**
Laslo Vespremi
**digital images:**
Paul Haeberli and Monica Schulze
**digital color separations:**
Kedie/Orent Inc.
**printing:**
Thomson Lithograph

**editorial:**
Silicon Graphics, Inc.
2011 N. Shoreline Blvd.
Mountain View, CA 94039-7311
415/335-1035
*<user@sgi.com>*

**advertising:**
Mkt.
210 Gilbert Avenue
Menlo Park, CA 94025
415/321-5312


## on the cover

*Kenneth Snelson's atom model is derived from the original Niels Bohr/Louis de Broglie matter-wave picture. His electrons are shown as standing waves which, like gross matter, are space-filling objects. The cover image refers specifically to the hydrogen atom. Despite containing but one electron, it has numerous auxiliary orbits from which to choose, depending on its energy state. Here, the electron's choices are shown for the first five levels, or shells. The colors of the ring-orbits mark those with identical numbers of de Broglie matter-waves. The "s" orbitals for successive shells are centered on the equator and are identical to the original Bohr/de Broglie model's orbits. All the rest (p,d,f...) are off-center, halo orbits. All one-wave orbits are blue; two-wave are yellow; three-wave are red; four-wave are green; and five-wave are purple.*

# contents



## about this issue

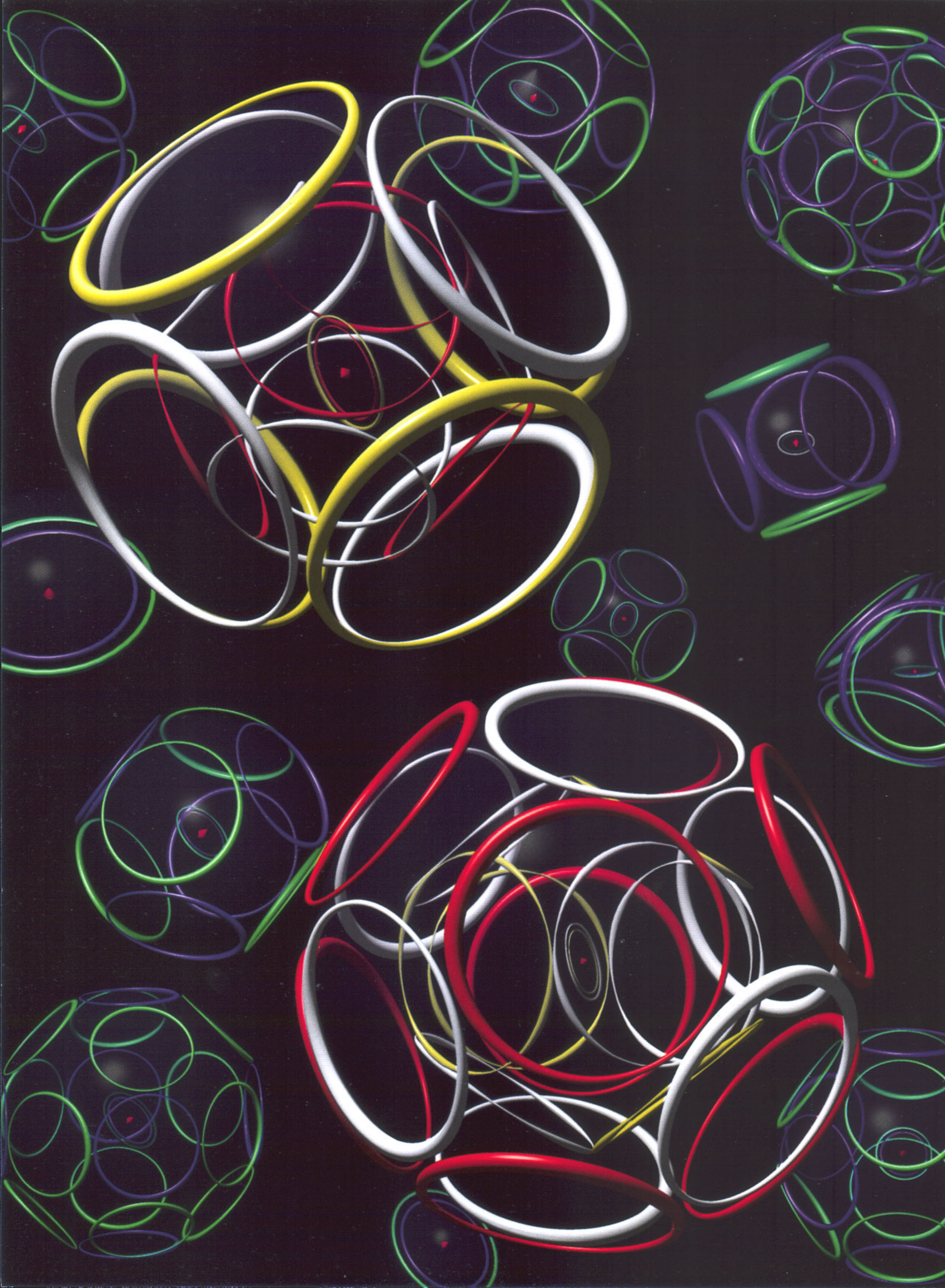
The benefits of visual computing seem intuitive, but this is not to say they're easy to demonstrate. Accordingly, Silicon Graphics has commissioned Peat Marwick Main & Co. to analyze the productivity gains available to the users of technical workstations through 3D. A final report is anticipated shortly, with the results scheduled to be published in the next issue of the *IRIS Universe*.

In the meantime, actual beneficiaries remain our best source for descriptions of the advantages inherent in visualization. It should come as no surprise, then, that this issue leads off with the tales of two users. One, sculptor/artist Kenneth Snelson, tells of the vistas now opening at the juncture of art and science thanks to the emergence of systems blending imaging tools with analytical ones. Mathematician Michael Mascagni follows with a description of the impact visualization has had on the development of mathematical models of the nervous system.

As the stories of both of these gentlemen attest, seeing truly *is* believing.

Mark Compton

The *IRIS Universe*, published quarterly by Silicon Graphics, Inc., is a publication dedicated exclusively to the needs and interests of the IRIS user community. Please send address changes to "IRIS Universe, Silicon Graphics, 2011 North Shoreline Boulevard, P.O. Box 7311, Mail Stop 130, Mountain View, CA. 94039-7311". Copying any portion of this publication for other than personal or internal reference purposes without the prior written permission of Silicon Graphics, Inc. is prohibited. Subscriptions are available upon request: write to the above address or call 415/962-3320. Correspondence regarding editorial (press releases and product announcements) should be sent to "Editor, IRIS Universe, 2011 North Shoreline Boulevard, P.O. Box 7311, Mail Stop 130, Mountain View, CA 94039-7311". Phone: 415/335-1035. E-mail: *<user@sgi.com>*. Letters to the *IRIS Universe* or its editors become the property of the magazine and are assumed to be intended for publication, and may be so used. Correspondence should include the writer's full name, address, and telephone number. All letters may be edited for purposes of clarity and/or space. "Silicon Graphics" and the Silicon Graphics logo are registered trademarks of Silicon Graphics, Inc.; "IRIS Universe", "IRIS 4D", "Power Series", "Personal IRIS", and "IRIX" are trademarks of Silicon Graphics, Inc. "UNIX" is a registered trademark of AT&T.



# structure, sculpture, and simulation: an interview with Kenneth Snelson

*By Frank Dietrich*

*More than ten years ago, I had my first encounter with Kenneth Snelson's sculptures. On display at a show in Berlin's National Gallery, a huge Mies van der Rohe building constructed of glass and steel, his complex works stood out dramatically against the simplicity of the environment. Later, while walking on the campus of Stanford University, I came upon another of his sculptures, once again riveting against a blue California sky.*

*Given the technological sophistication of Snelson's work, I was not surprised to learn recently that Snelson had started to work with an IRIS workstation. But, recognizing that the creation and manipulation of computer-generated objects rather than tangible ones requires a substantial mental shift, I immediately became curious about what had prompted Snelson to make the jump.*

*One of the strongest influences on Snelson's work, it turns out, was a summer spent more than 40 years ago at Black Mountain College (located close to Ashville, NC), where Buckminster Fuller and a number of other American avant-garde artists, writers, and performers were teaching, together with various exiled Bauhaus members. It was after this experience that Snelson invented a new structural form in his 1948 sculpture, "Early X Piece". Fuller was so intrigued when shown the work that he later coined a word for Snelson's discovery: "tensegrity", a contraction of tension and integrity. Tubular parts in "Early X Piece" maintain Snelson's signature tension by pushing outward against a closed network of wire members. Every component of the assembly is essential to maintaining balance.*

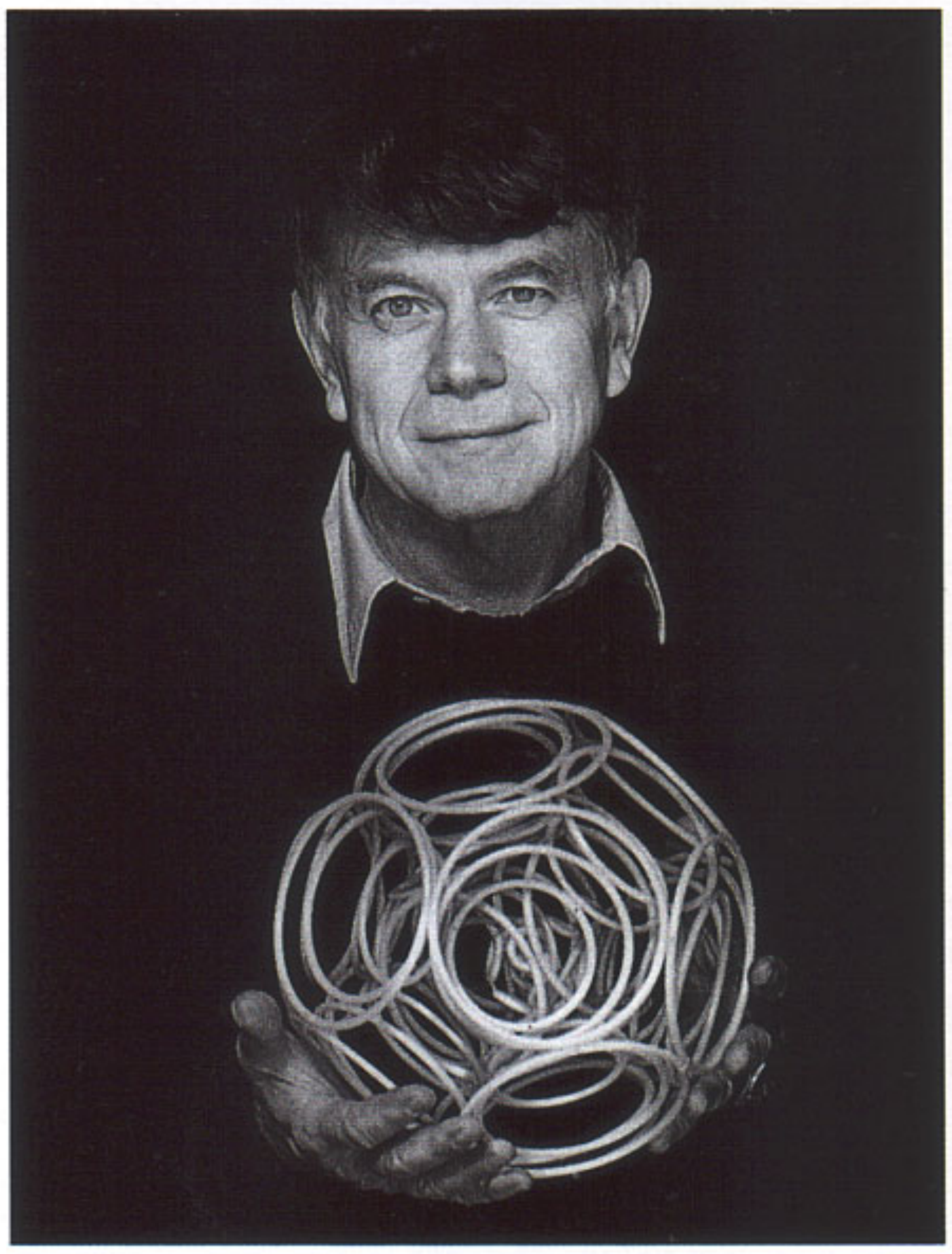


PHOTO: ANDREA SNELSON

*Tensegrity has continued to be an important theme for Snelson. His work has also been greatly influenced by an abiding interest in atomic structures. For nearly three decades, in fact, Snelson has been working visions of atomic matter wave into many of his models.*

*Snelson's sculptures and images can be seen worldwide in museums and in corporate and private collections. Currently, a Snelson exhibition of computer-generated images is at New York's Academy of Sciences (through April 7). Afterward, the show will travel to the California Museum of Science and Industry in Los Angeles' Exhibition Park (June 9 - August 27) and to the National Academy of Sciences in Washington, D.C. (April 5 - June 25, 1990).*

**IRIS** Many artists today are trying to make sense of computers. What has your IRIS done for you?

**SNELSON** For one, it has kept me in my studio 14 hours a day, seven days a week, for the last ten months.

**IRIS** So you like it?

**SNELSON** I do, even though I find it to be frustrating at times. I used PCs for word processing and paint programs for four years but that was minor league stuff compared to what I'm doing now. In fact, I've been amazed to learn just how complex UNIX can be. Even after these many months I still haven't learned it all that well, but I *can* manage.

**IRIS** I'm extremely impressed by the quality of the images you've been able to produce after such a short amount of time. You're also quite prolific.

**SNELSON** Well, it's hard to imagine anyone spending more time using a workstation than I do -- unless they were being paid well for it.

**IRIS** Your images have a distinctive structure -- one that escapes most other artists. What is it that you are trying to accomplish?

**SNELSON** I am trying to capture images that I've longed to make for some time but haven't been able to produce by any other means. I do believe, though, that artists ought to have a compelling reason for choosing to make their statements via computer rather than through some other medium, such as painting, theater, or whatever. Of course, there are a lot of ways in which artists now use computers to do something other than simply produce pictures. But as far as images are concerned, I think many of the pictures coming out of scientific computing today are more engaging than those intentionally developed as works of art. I'm speaking of, say, a picture depicting air flows over an airfoil. Images of that sort can be quite startling.

**IRIS** I agree. They tend to have a higher degree of complexity than images made up wholly of geometric functions. Secondly, the kind of motion you find in many scientific images can be quite unusual. That's not surprising, I suppose, given that the pictures depict a domain that's not part of our visual world. Bringing out into the open that which has not previously been visually available is very intriguing -- which leads me to ask: have you found the IRIS visualization tools to be well suited to your purposes?

**SNELSON** They're quite well suited to what I've wanted to do for many years, which is to make images of the atom. Since 1960, I've evolved a different interpretation of how electrons occupy the atom, giving it structure and form. It's a new and helpful method for visualizing atoms. And, believe me, I've tried in every imaginable way to present these ideas: I've written about them, I've painted this picture in my mind, I've made models from all kinds of materials...you name it. At one time, I thought I might be able to accomplish the effect I wanted with an airbrush, but I found I wasn't skilled enough as an airbrush artist. So I hired a top graphics artist to render pictures of my invisible, ethereal, majestic atom. Over a period of several weeks, he produced ten or twelve of the pictures I sketched. Then, when we got to the really complex ones, he gently advised me that the cost was going to get outrageous. So, reluctantly, I had to shelve the project -- permanently. Also, of course, I found it awkward to hire another artist to execute images I couldn't realize on my own, but I had no hope of ever becoming a *virtuoso* with the airbrush.

That was ten years ago. Now, with my workstation, I can produce just the sort of images I was looking for -- with more complexity and subtlety than an airbrush artist could achieve in a decade. This, of course, is arguable since no computer can yet do everything the hand can. Still, for the kind of geometrical, three-dimensional pictures I want to produce, the computer works amazingly well. It enables me to make images of an invented world and to capture worlds that may exist even if they're invisible.

**IRIS** Can the system do everything you want?

**SNELSON** Mostly. Yet, there are limitations. For instance, while the orbital pathways of my atomic pictures are meant to act like space-filling matter, I'd still like them to appear more like glowing vapors than objects made of plastic or felt. That's hard when you're working with polygons. The pathways ought to look something like neon, but cause refraction when you're looking through them. Also, I wish there were no limits on the number of polygons that can be used in a picture. But, polygons eat up memory and there never seems to be enough. Another thing is that rendering still is unconscionably slow when it comes to ray tracing and shadow casting and all those amenities.

But there is a lot of visual magic in the IRIS. One of my greatest joys is seeing my images on the monitor stereoscopically. I've been interested in 3D photography since I was a child. And, as you know, in the wireframe stages of developing computer images, it often is hard to tell what's in back of what. So, in Wavefront *Preview*, I've set up two small camera ports, with the right-eyed view on the left and the left-eyed view on the right -- like a reversed stereo pair. You have to be able to "free-view" cross-eyed, though, fusing the two images as many stereophiles can. Also, seeing objects in 3D in real-time has helped me enormously whenever I've needed to set my objects on the *z*-axis.

**IRIS** You bring a wealth of experience to bear when you design tensegrity into a structure. Would it be difficult to formalize those considerations such that they could be implemented by the machine?

**SNELSON** It might be possible. I'd like to think so, anyway. But, for the present at least, it's still a pretty awkward exercise to use computer modeling to establish each compression member in space before finding the exact points where polygons should be connected to represent a stretched tension wire. I think a program would have to be specifically designed to accomplish that with ease. Unfortunately, I'm as unlikely to become a *virtuoso* programmer as I am to become an airbrush artist.

**IRIS** The solution you're referring to is one in which everything would have to be interconnected, and -- I assume -- programming these interconnections would probably be quite difficult. Do you know from experience how several parameters of your structures influence each other?

**SNELSON** Oh, of course. In the process of making hundreds of sculptures, I've learned that empirically. There are general rules: a structural scheme that controls all necessary directions. It's analogous to setting down the legs of a tripod one by one. A tripod doesn't need a fourth leg, and -- just so -- none of the wires in one of my sculptures is redundant in that all are required to keep everything in place. As for figuring out how to accomplish this by computer, I can't imagine how one would get around the hard job of building the structure out of metal to experience how it works. Still, given the principles, it seems that a program could be written to adjust everything in space just as water finds its own level in a pond. Even so, since the right stresses in a sculpture are arrived at in much the same way that a violin is tuned, I'm not convinced there would be all that much to gain by using computer models.

**IRIS** The fact that you've been able to produce the images you have demonstrates just how far we've already advanced in terms of making this technology accessible.

**SNELSON** Yes, it says a lot for the computer. I remember seeing workstations at SIGGRAPH in '84 and asking people, "How much does one of these systems cost?" "About a quarter of a million," was the answer. I figured it would be *years* before I could own one. I was astonished to find only three years later that I could buy a system for about a third of that amount. Still, few artists can afford one today, nor are many crazy enough to try -- unless, of course, they're prepared to mortgage their lives and forget about sending their kids to college.

As IRIS workstations become even more affordable, I think a lot of people are likely to find it irresistible to own such an incredibly valuable art tool. When that time arrives, I only hope we'll all have an easier means than we currently do for obtaining photo-quality hardcopy. Presently, that's the major headache. When better output devices become readily available -- and when Silicon Graphics and/or some other company comes out with a system that encourages software vendors to make their products available at more popular prices -- you're going to see whole new worlds of use open up for these systems. It's inevitable.

*Frank Dietrich formerly was editor of the IRIS Universe.*

# animation's role in mathematically modeling the nervous system

*By Michael Mascagni*

The history of mathematically modeling the nervous system extends almost to the very beginnings of the field of electrophysiology. In fact, the first crude electrical recordings of neuronal activity were interpreted using simple models based on electrical cable properties of the neuron. Mathematically, these linear diffusion models had qualitative and quantitative properties that one could describe using paper and pencil [*Jack75, Pesk76*]. For example, the equation for determining the voltage on an electrically passive cable is:

$$C\frac{\partial V}{\partial t} = \frac{a}{2R}\frac{\partial^2 V}{\partial x^2} - \bar{g}V.$$

This equation is a linear partial differential equation of the parabolic type, and is derivable by applying Ohm's law and current conservation for a leaky cable with a uniform radius. But while the scientists who performed the earliest recordings of neuronal activity could use these models to successfully describe certain aspects of the electrical properties of nerve cells, they could not adequately explain some of the most striking behaviors observed in nerves.

Among the mathematically elusive behaviors was the initiation and propagation of the so-called "action potential", the wave-like disturbance in the electrical potential across a neuron's cell membrane that occurs when a nerve cell is electrically stimulated. This pulse-shaped voltage wave propagates down the length of the neuron with a nearly constant velocity, almost without a change in shape. The physiological significance is that this nonattenuating wave form serves as the basis of high-fidelity information transmission over long distances within the nervous system.

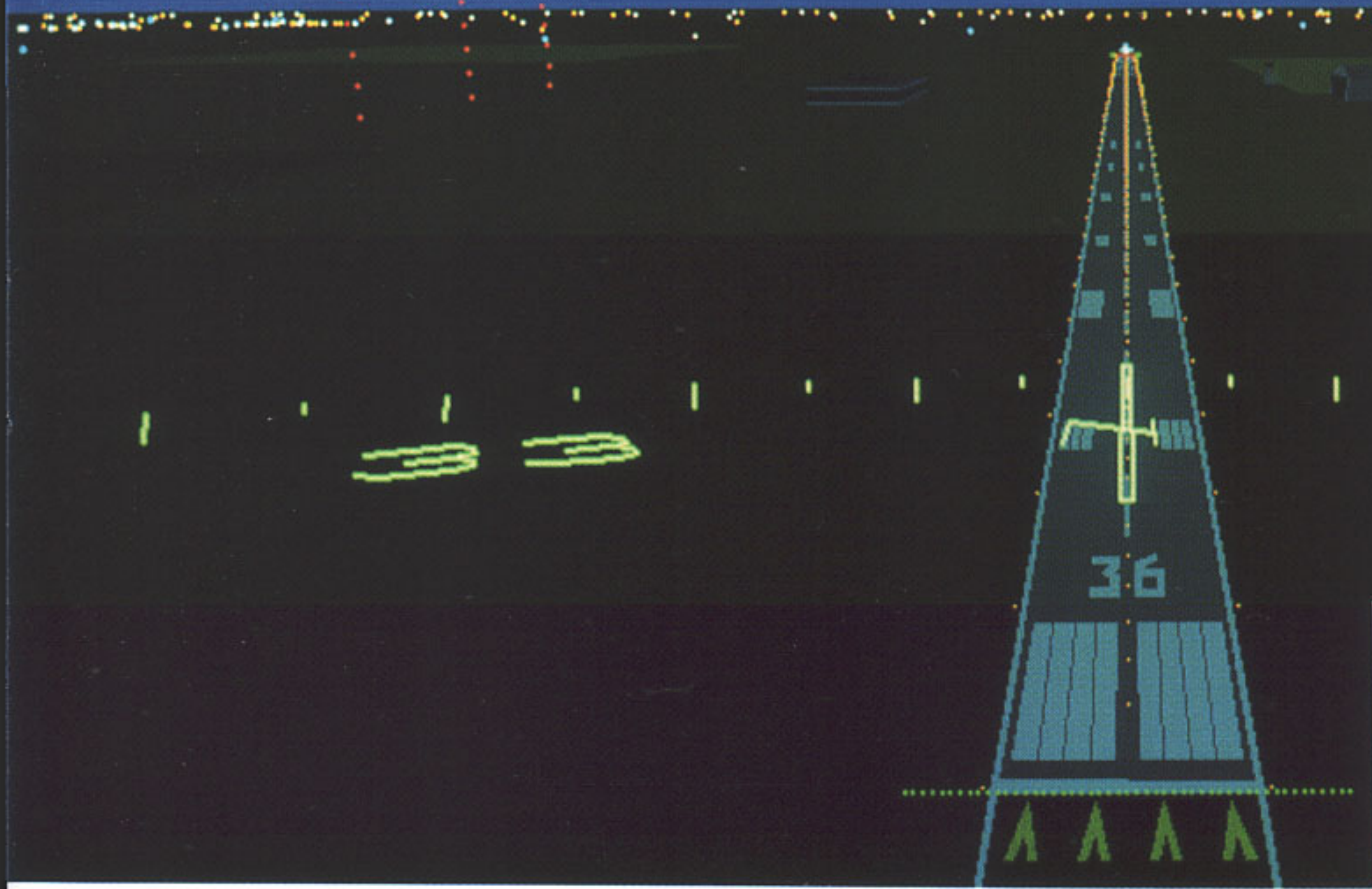
The first model to adequately describe the action potential was a phenomenological one derived by British scientists Hodgkin, Huxley, and Katz from careful experimentation on a giant axon found in a particular species of North Atlantic squid. The model (which ultimately earned the scientists a Nobel Prize) was based on mathematically similar diffusion models, but included nonlinear terms describing the time course of currents across the axon membrane. The nonlinear parabolic partial differential equations for this model of nerve conduction are:

$$\begin{aligned}
C\frac{\partial V}{\partial t} &= \frac{a}{2R}\frac{\partial^2 V}{\partial x^2} - \bar{g}_{Na}m^3h(V - E_{Na}) - \\
&\quad \bar{g}_K n^4(V - E_K) - \bar{g}_L(V - E_L), \\
\frac{dm}{dt} &= (1 - m)\alpha_m(V) - m\beta_m(V), \\
\frac{dh}{dt} &= (1 - h)\alpha_h(V) - h\beta_h(V), \\
\frac{dn}{dt} &= (1 - n)\alpha_n(V) - n\beta_n(V),
\end{aligned}$$

where the $\alpha$ and $\beta$ functions are:

$$\begin{aligned}
\alpha_m(V) &= \frac{\frac{(25 - V)}{10}}{e^{\frac{(25-V)}{10}} - 1}, \ \beta_m(V) = 4e^{-\frac{V}{18}}, \\
\alpha_h(V) &= 0.07e^{-\frac{V}{20}}, \ \beta_h(V) = \frac{1}{e^{\frac{(30-V)}{10}} + 1}, \\
\alpha_n(V) &= \frac{\frac{(10 - V)}{100}}{e^{\frac{(10-V)}{10}} - 1}, \beta_n(V) = 0.125e^{-\frac{V}{80}}.
\end{aligned}$$

# VISUAL STIMULATION



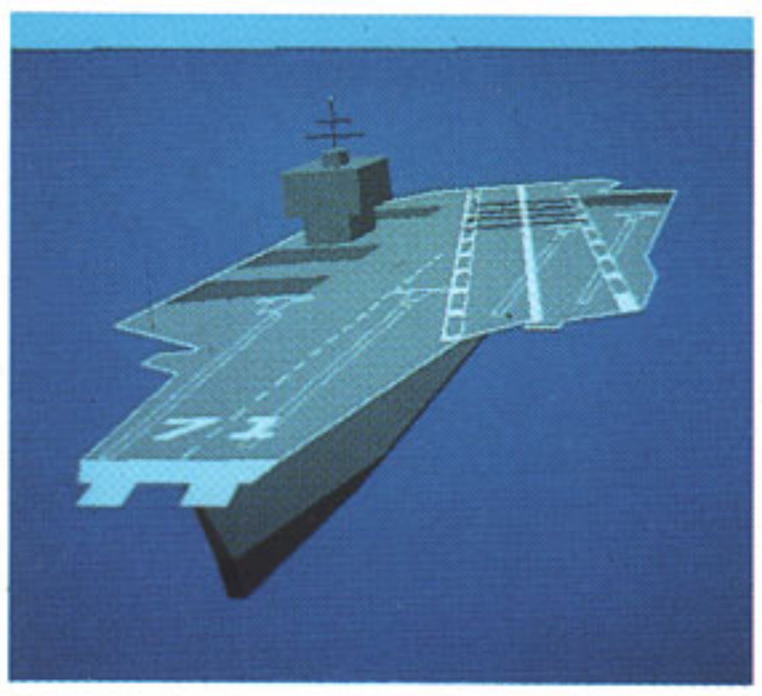

## *ONE SOFTWARE TOOL FOR ALL YOUR REAL-TIME VISUALIZATION NEEDS*

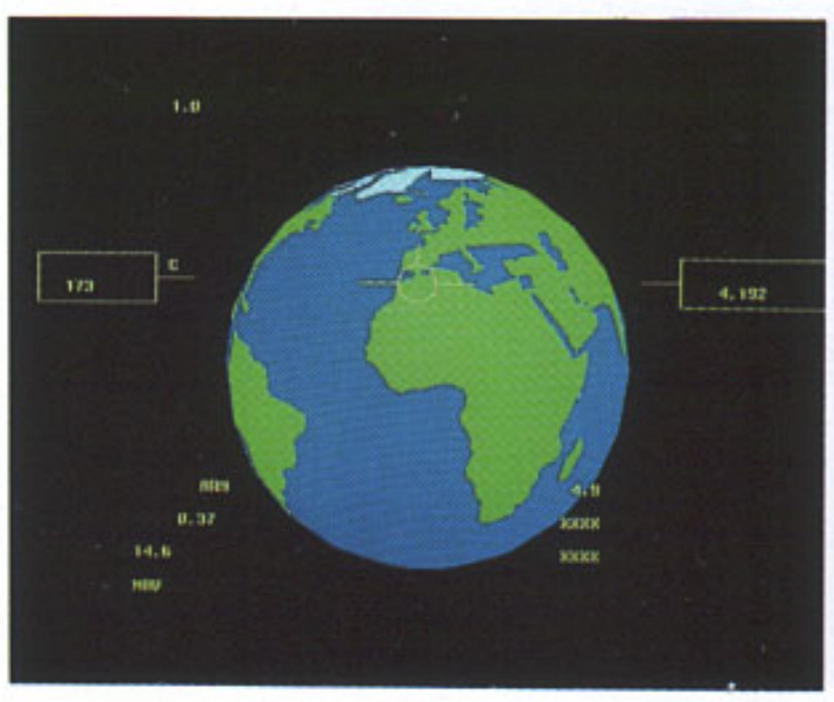
*What do British Aerospace, McDonnell Douglas, TRW, Rockwell International, Northrop, the U.S. Navy, the U.S. Air Force, and other high-technology organizations have in common?*

### THE GENERIC VISUAL SYSTEM™

**GVS** software provides a powerful and cost-effective solution for embedded systems and computer generated imagery (CGI) applications. The GVS provides portable, real-time solutions for any graphics application, e.g., flight, ground, or space simulation, which requires fast display update rates.

- Create your own universe easily — add your own out-the-window scenes or 2-D and 3-D symbology applications
- Add your own special effects or use ours
- Supports your single or multiple channel applications
- Easily interface your external host computers — choose Ethernet TCP/IP, UDP, or shared memory network hardware available from Gemini Technology
- Interfaces with your *existing* simulation software
- Maintains graphics hardware operation at peak performance
- Automated level of detail scene management
- Automated airport scene generation tools
- 3-D object control is as simple as X-Y-Z, pitch-roll-heading
- Data bases—model directly with the GVS or import data base models from various off-line interactive modelling systems including MULTIGEN or SUPER 3D
- Hardware independence—PHIGS, PHIGS + compatible
- Rapid prototyping for real-time symbols including Head-Up, Multi-Purpose, or Helmet Mounted Displays
- Software development with Fortran-77, C, or Ada®






GVS

For information or a demonstration,

**Gemini Technology Corp.**
11770 Warner Avenue, #116
Fountain Valley, CA 92708 USA
***Tel:* *(714) 966-8455***
***Fax:* *(714) 540-5261***

Due to the nonlinear ionic currents of the model, insight into its behavior could not be derived from closed-form analytic solutions. It thus became clear that nonlinear diffusion equations of this sort would have to be explored *numerically* for their various quantitative and qualitative features. In fact, in the paper where the Hodgkin-Huxley model first appeared in its entirety, numerical computation of the solution was used to provide a basis for comparing the model to the experimental data [*Hodg52*].

## modern developments

Since the introduction of nonlinear diffusion models to neurophysiology, a considerable amount of computation has taken place to uncover properties of the Hodgkin-Huxley equations and other related models. Up until now, though, most of these computations have been restricted in scope to examinations of questions about single neurons and how these models of single units replicate the behavior of a variety of real neurons known well to the experimentalists [*Dodge79, Dodge73, Traub77*] (see Figure 1).

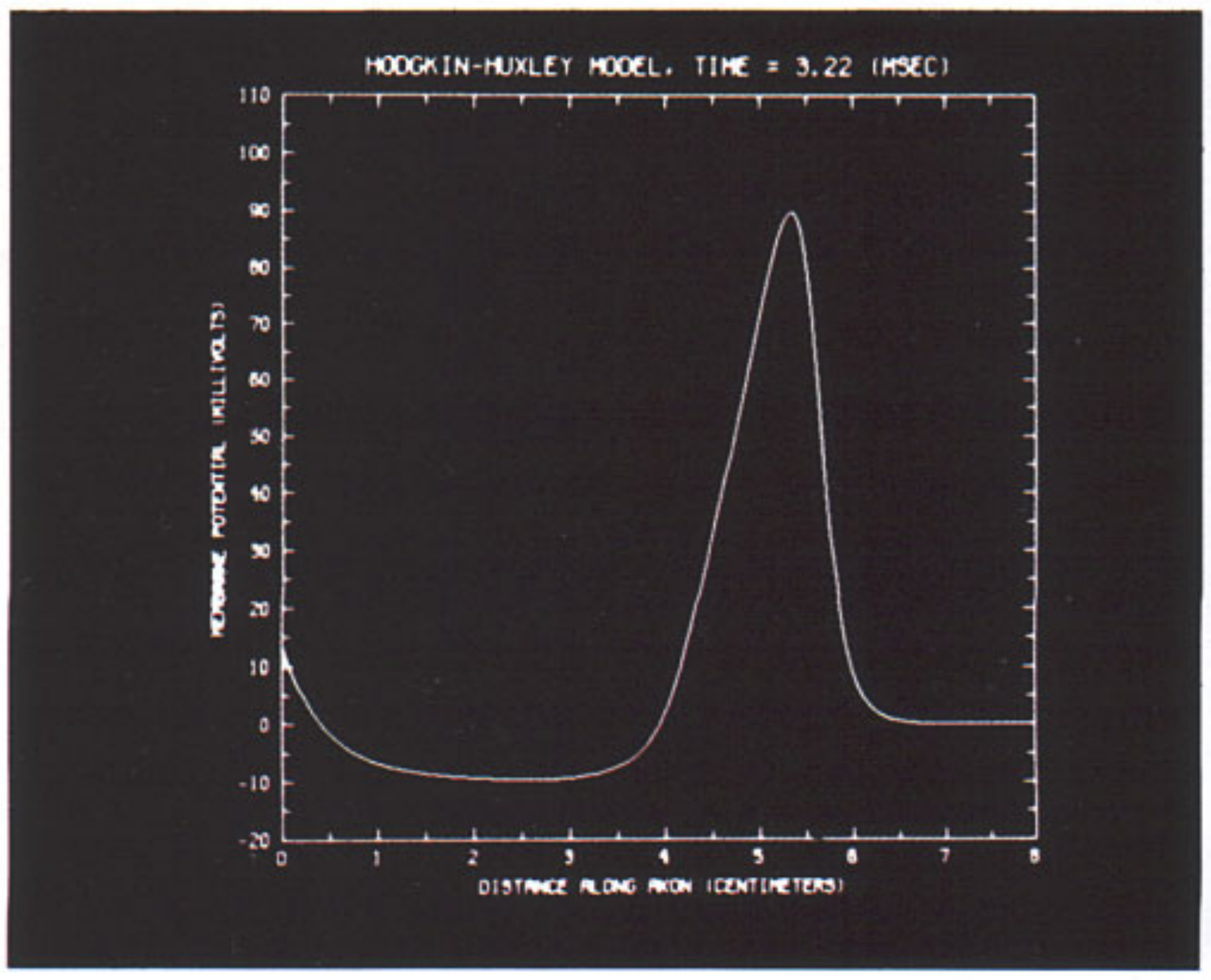


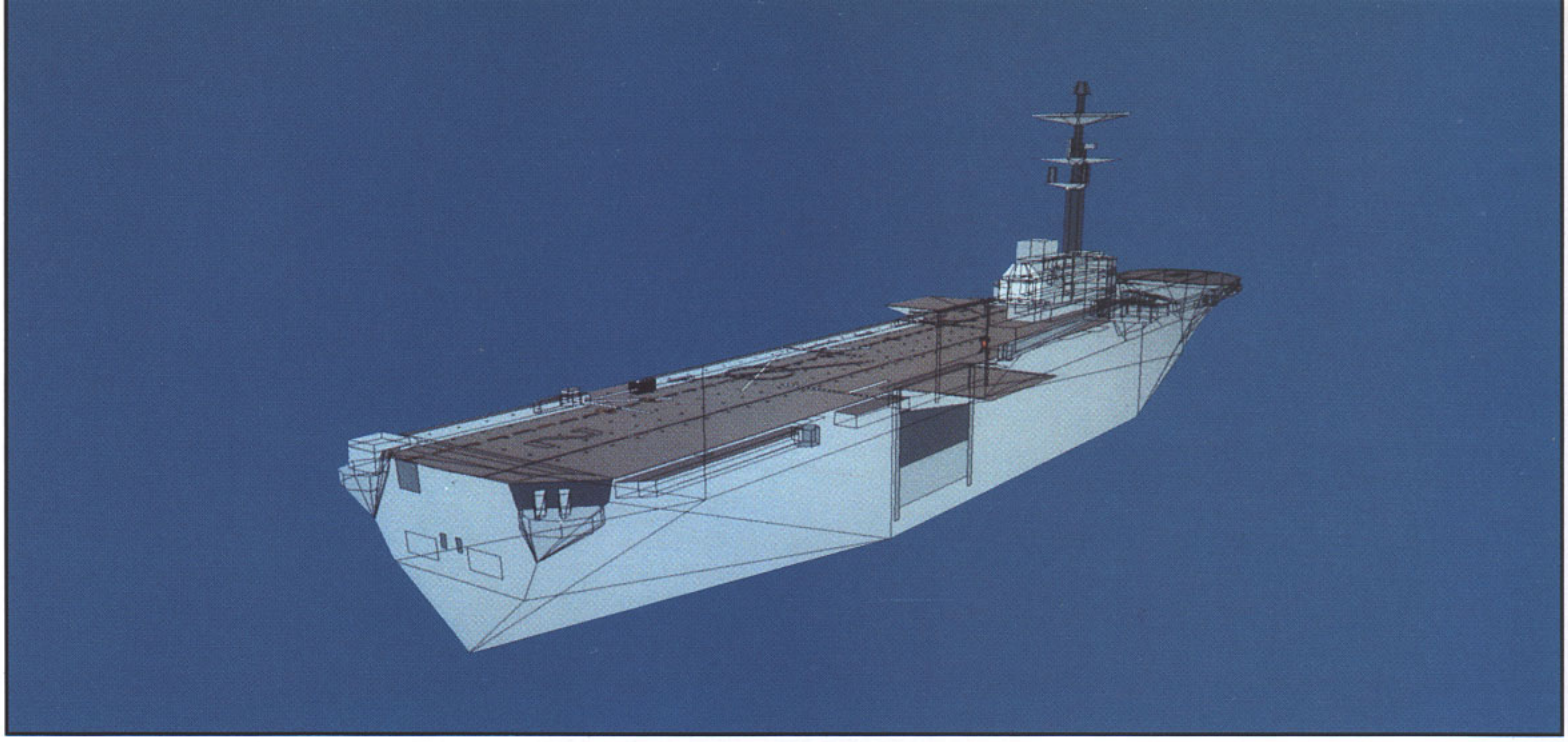

***Figure 1:*** *Plotted values of membrane potential versus distance for a single axon with Hodgkin-Huxley kinetics. The axon was stimulated at the left-most point with a constant injection of stimulating current, causing the initiation of action potentials, which propagate from left to right. (Image produced with the help of Ed Friedman and Jeff Bary of the Academic Computing Facility at New York University's Courant Institute.)*

The numerical solution of these model equations has historically required considerable computer time. This has restricted computational investigation to questions about single neurons or events that manifest themselves rapidly at the onset of action potential activity. Now, with the advent of modern vector-processing supercomputers and new refinements in algorithms for the numerical solution of several Hodgkin-Huxley-type equations, these computational constraints no longer are significant. In recent years, questions having to do with long-term behavior of neuron models, as well as those having to do with the interaction dynamics of neurons linked together in networks, have been addressed [*Masc87, Masc89*]. Also, in somewhat related work, the behavior of large ensembles of neurons has been studied using model neurons with greatly simplified dynamics [*Sejn88*].

My own work has focused on the simulation of a moderate number of neurons that can be coupled in rather arbitrary ways. The primary motivation in this is the hope that the kind of detailed information available during the simulations of small pieces of neural tissue will both adequately replicate known electrophysiological data and illuminate interesting phenomenology that, until now, has not been observable. It must be emphasized that to obtain intracellular voltage recordings from real neurons is an extremely time consuming and technically demanding undertaking (not to mention one that usually only obtains voltage versus time data at one point in a single neuron).

A numerical simulation, on the other hand, provides access to the entire history of every dynamic quantity at every point along all of the modeled neurons. It requires only a modest leap of faith to believe that even if one adequately simulates an already well-known phenomenon, the availability of extra data will surely provide additional insight into the system's behavior.

There are, however, many significant challenges involved in producing realistic models representing groups of interacting nervous system cells. By far the most daunting are the impediments to experimentally determining the electrical constants and nonlinear ionic current-voltage relationships in all of the neurons of interest. Unfortunately, the modeler must wait for the experimentalist to provide these data, and the wait can be long since this information often is not critical to the experimentalist's own research agenda. The technical

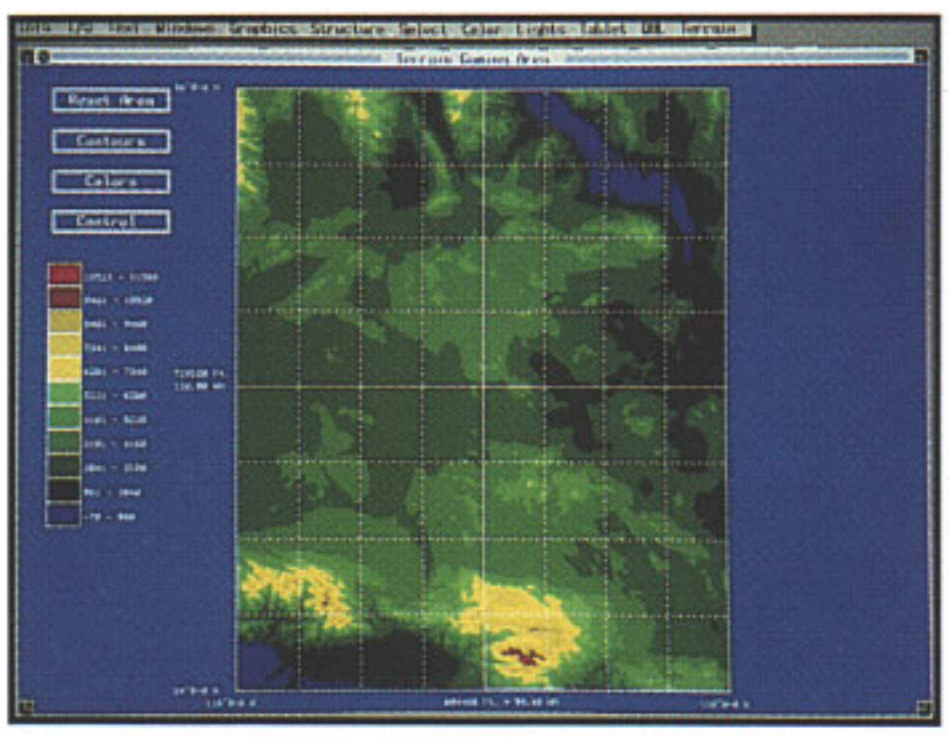


# *Would you like to create visual scenes in hours instead of weeks?*

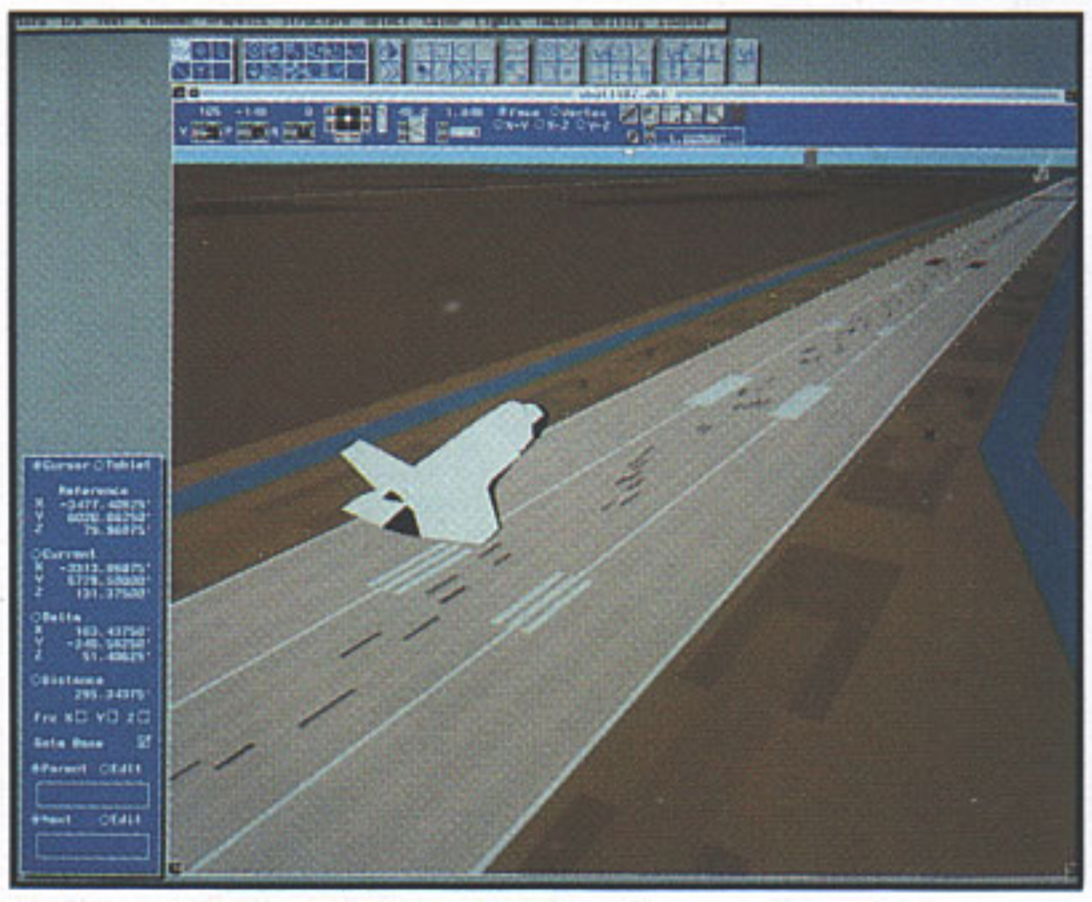
***MultiGen*** ™ *gives you a fast, easy way to create and modify visual simulation data bases.*

MultiGen is a graphical editor for creation, modification and viewing of visual simulation scenes. Rapid, precise, WYSIWYG style data base modeling is made possible by an intuitive graphic interface and a rich set of 3D modeling tools designed specifically for visual simulation.

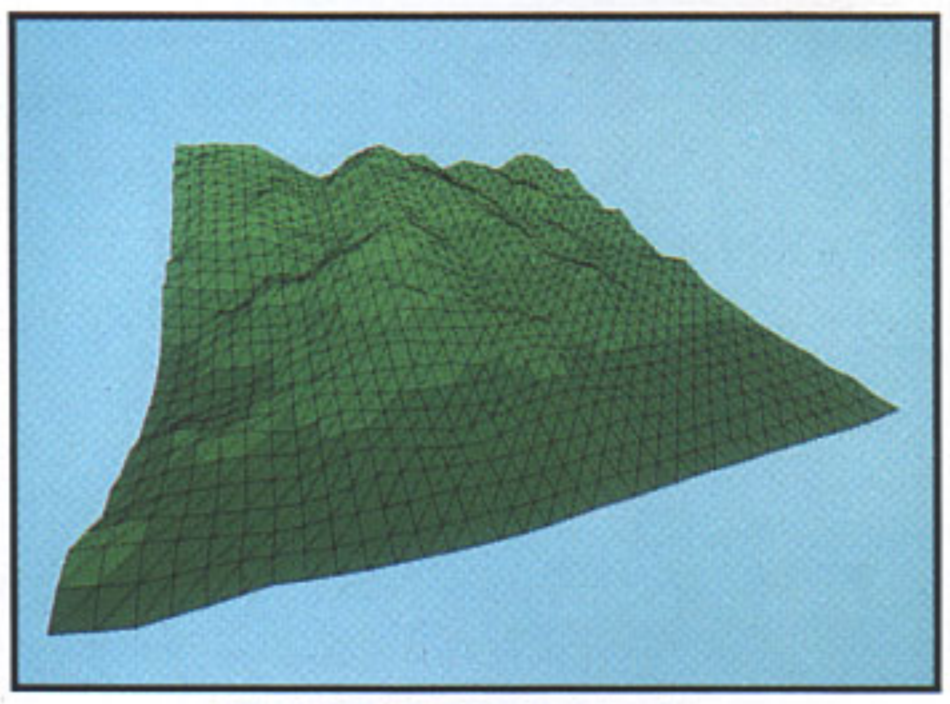
Software Systems developed MultiGen to address the requirements of both high-end image generators, as well as cost effective, IRIS-based real time systems. An elegant, public domain data base format (called Flight) is now available that is ideally suited for low to medium fidelity visual systems. Further, since data bases developed with MultiGen can be stored in a standard format, it is now possible to transfer models between different visual systems supported by MultiGen's Import/Export option.



**DMA DTED contours**



**Extensive modeling & viewing controls**



**DMA DTED 3D display**

The DMA DTED conversion option automatically generates 3D polygonal terrain from DMA elevation data. This terrain may then be populated with various objects and examined with MultiGen's assortment of viewing controls (including a fly-through capability), or "flown" with a compatible real-time system.

MultiGen integrates a set of powerful tools, options and graphical interfaces to make visual scene modeling fast and easy. It is available on the complete line of Silicon Graphics workstations.

Call to obtain sample data bases and a specification of the Flight visual data base format.


**Software Systems** *1625 The Alameda, Suite 700, San Jose, California, 95126 • Telephone 408-995-0689 • Facsimile 408-995-5058*

obstacles within the modeler's own sphere include the development of mathematical software for the efficient computation of solutions to modeled systems, and the production of graphical software to aid in the interpretation of the simulated results.

## supercomputers and graphics

The technical problem of efficiently and rapidly computing solutions to model equations has been adequately solved with the help of a Cyber 205 at the John von Neumann National Supercomputer Center (JvNC), and a Cray X/MP at the National Cancer Institute's Advanced Scientific Computing Laboratory (ASCL). Codes that compute the time evolution of voltages on an arbitrarily connected network of neurons at speeds only 100 times slower than real time have been optimized on the two supercomputers. Given this advance, it seems evident that, with the ever increasing computational power of state-of-the-art supercomputers and the increased availability of parallel computers, close to real-time simulations of model equations including as many as a few thousand neurons will be possible in the very near future.

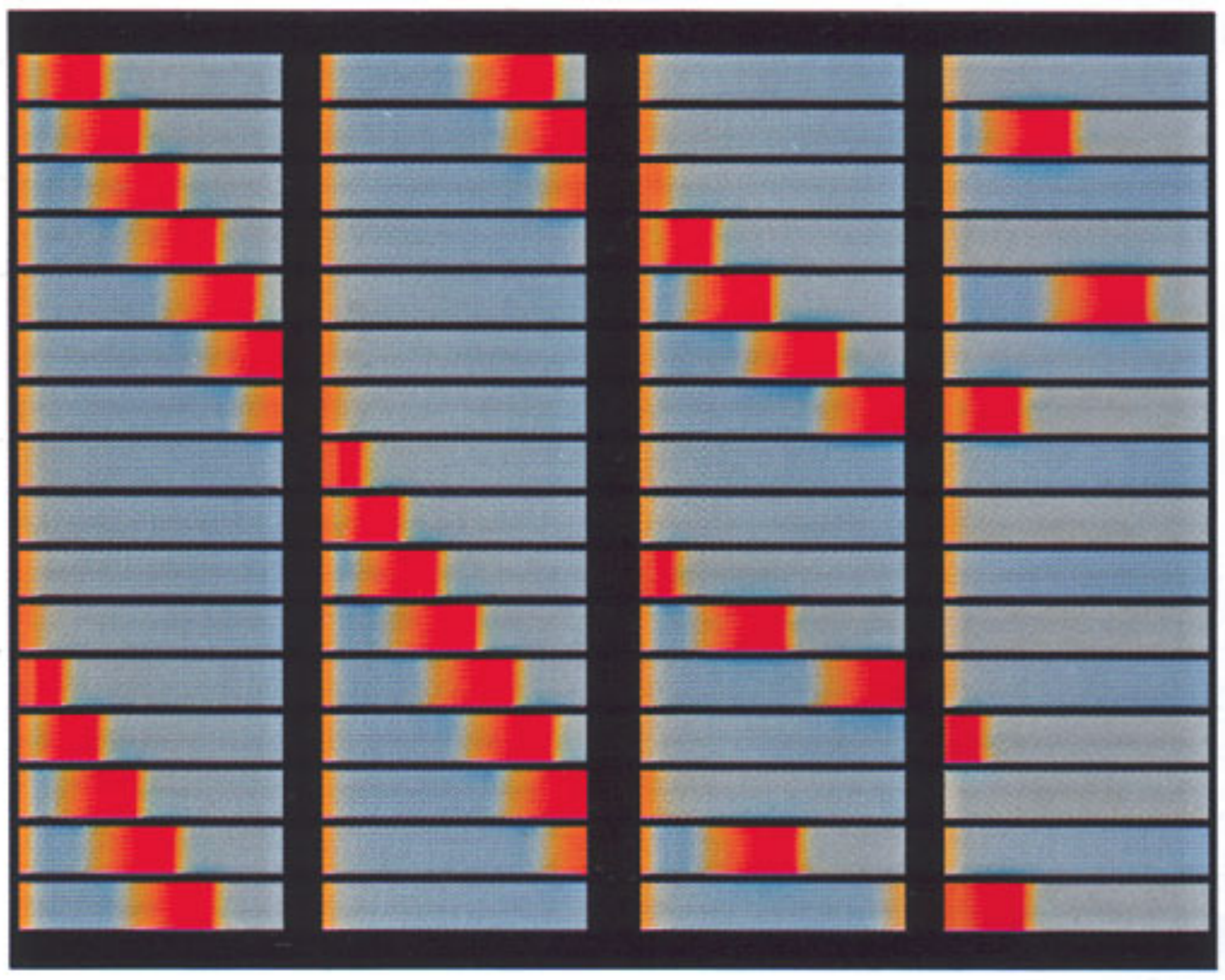


***Figure 2:*** *Voltage versus distance is plotted for an ensemble of 64 neurons with passive dendritic regions and active axonal regions using Hodgkin-Huxley kinetics. The neurons are linked end to end in an overall ring topology with units numbered from the top down and from left to right. Here the voltage information is encoded in a colormap related to the Boltzman black body spectrum. This particular time step is part of a 4000 millisecond simulation computed on a Cray X/MP at the National Cancer Institute's Advanced Scientific Computing Laboratory in Frederick, MD. The image was taken from the console of the IRIS 4D/60T used to perform the rendering. (Tom Palmer of the Advanced Scientific Computing Laboratory aided the visualization.)*

The problem of interpreting the massive amounts of data potentially available from these supercomputations, though, remains most daunting. Providing for a viewing of all the data produced in even a short computation is a staggering undertaking, and deciding on a rationale for selecting and displaying the most significant data is an open problem with implications for the experimentalist as well as the modeler. Accordingly, I have chosen a fairly technologically intensive compromise to the problem: instead of deciding beforehand which data are important and which are not, I have chosen to view the entire time history of the voltage on the full network of neurons. For the computations currently of interest to me, this means that at each time step I have 201 (spatial grid points per neuron) X 64 (number of neurons) = 12864 voltage values to display. My preliminary viewings of these data (in the form of graphical animations) have occurred on an IRIS 4D/60T at the ASCL. Then, upon achieving a computational result I wish to document, I've worked with the visualization group at the University of Illinois' National Center for Supercomputing Applications (NCSA) to obtain a videotape "hardcopy" of the computations.

In Figure 2, a still photograph shows a single time step of a 64-neuron computation, as visualized on the 4D/60T. Here, the colormap, which is related to the Boltzman black-body spectrum, encodes the voltage values. The blues and greens signify low voltage values, while the reds, oranges, and yellows indicate high voltage. In Figure 3 is a still from a videotape made using the same data. Here, though, the ring structure of the neuronal connectivity is more readily apparent, and voltage is encoded not only by color but also by the physical height of the voltage polygons. Figure 2 was rendered on the 4D/60T using SGI's FORTRAN-callable graphics routines, while the image in Figure 3 was produced on an Alliant FX/8 using Wavefront Technology's rendering package and later was transferred to film from videotape.

Figure 3 represents the efforts of a large, state-of-the-art scientific visualization facility. The quality of the final product is magnificent, but the resources required to produce this image are available only to those with access to NCSA or similarly equipped sites. The visualiza-

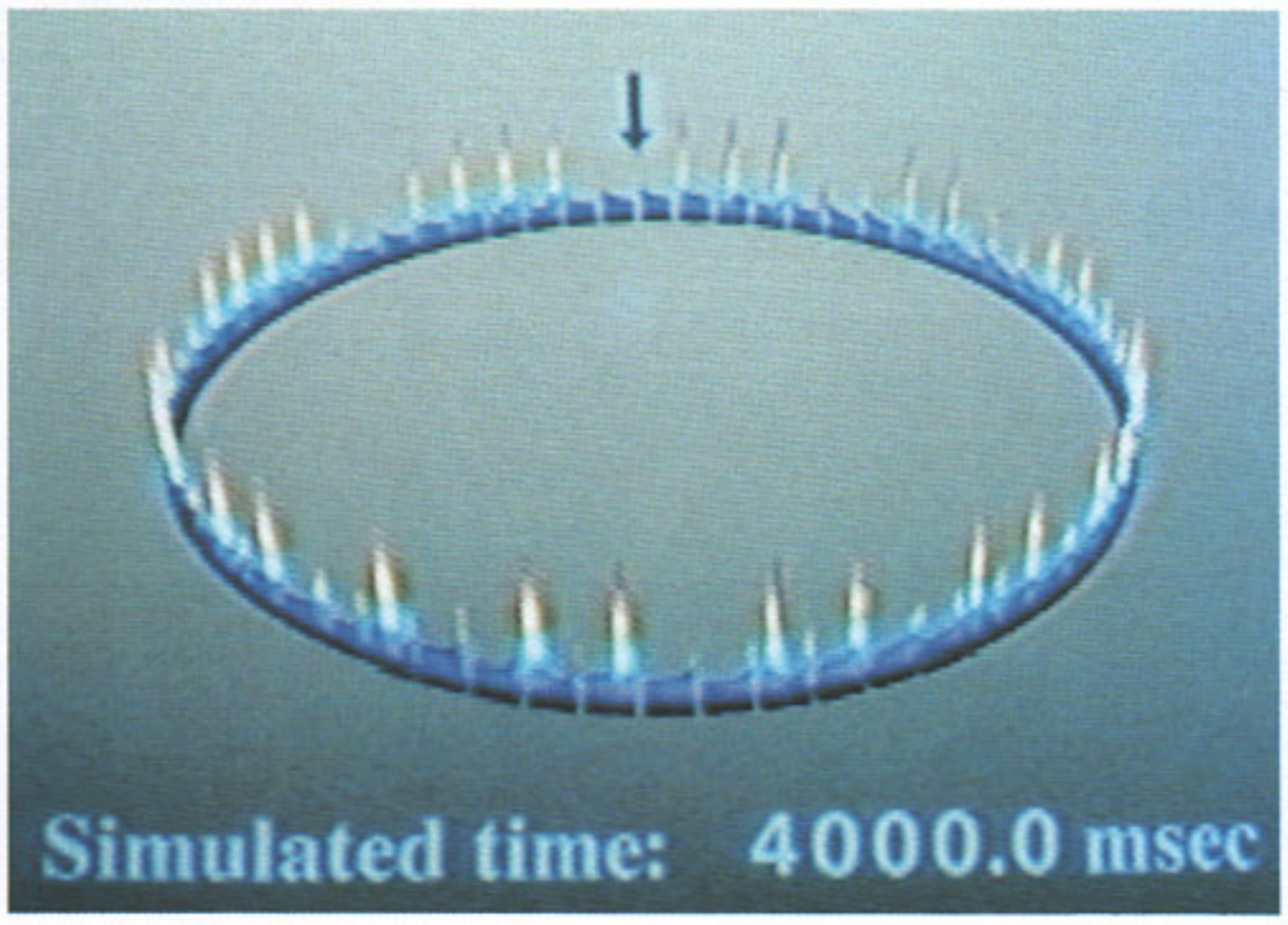
tion shown in Figure 2 is much more abstract and has less visual impact, but it also required only a very small time investment in terms of software development, and it provides the researcher with immediate access to the computational results. This is to say that all of the information conveyed in Figure 3 is contained in Figure 2, yet -- with no production lag time -- one could modify and fine-tune a mathematical model several hundred times with the aid of a visualization like that shown in Figure 2 before producing a final hardcopy of Figure 3's quality. Obviously, when one uses supercomputers to refine complicated mathematical models, access to both kinds of visualization is indispensible.

## a model problem

I have used a model problem to develop the computational and visualization techniques I believe to be essential to the modeling of actual pieces of the nervous system. The problem involves studying the behavior of a ring of 64 neurons, each with Hodgkin-Huxley kinetics. This ring of neurons is hooked together through connections modeling the synaptic connections known to exist in real neurons. The particular synapses on these neurons are purely excitatory, meaning that activity on one neuron will lead to an enhancement of activity in coupled neurons.

The ring is driven into activity by (mathematically) applying an external constant current source to a neuron on the ring. This constant current will cause the first neuron to fire action potentials at regular intervals. These then will propagate through the ring, one neuron at a time, and eventually come back to the first neuron. While this is only a test computation, a question of mathematical interest has to do with whether the return of the propagated impulses around the ring disrupts the overall regular pattern of activity that's initiated around the ring by the regular firing of the first neuron.

Two pictures tell us whether the point-stimulated ring of neurons develops periodic behavior. Figure 3 shows the ring just before the first action potential has propagated past its original point of initiation on the first neuron, and Figure 4 shows the ring well after the action potentials have gone around the ring several times. Very regular spatial patterns in Figure 3 extend over the length of several neurons, a clearly periodic



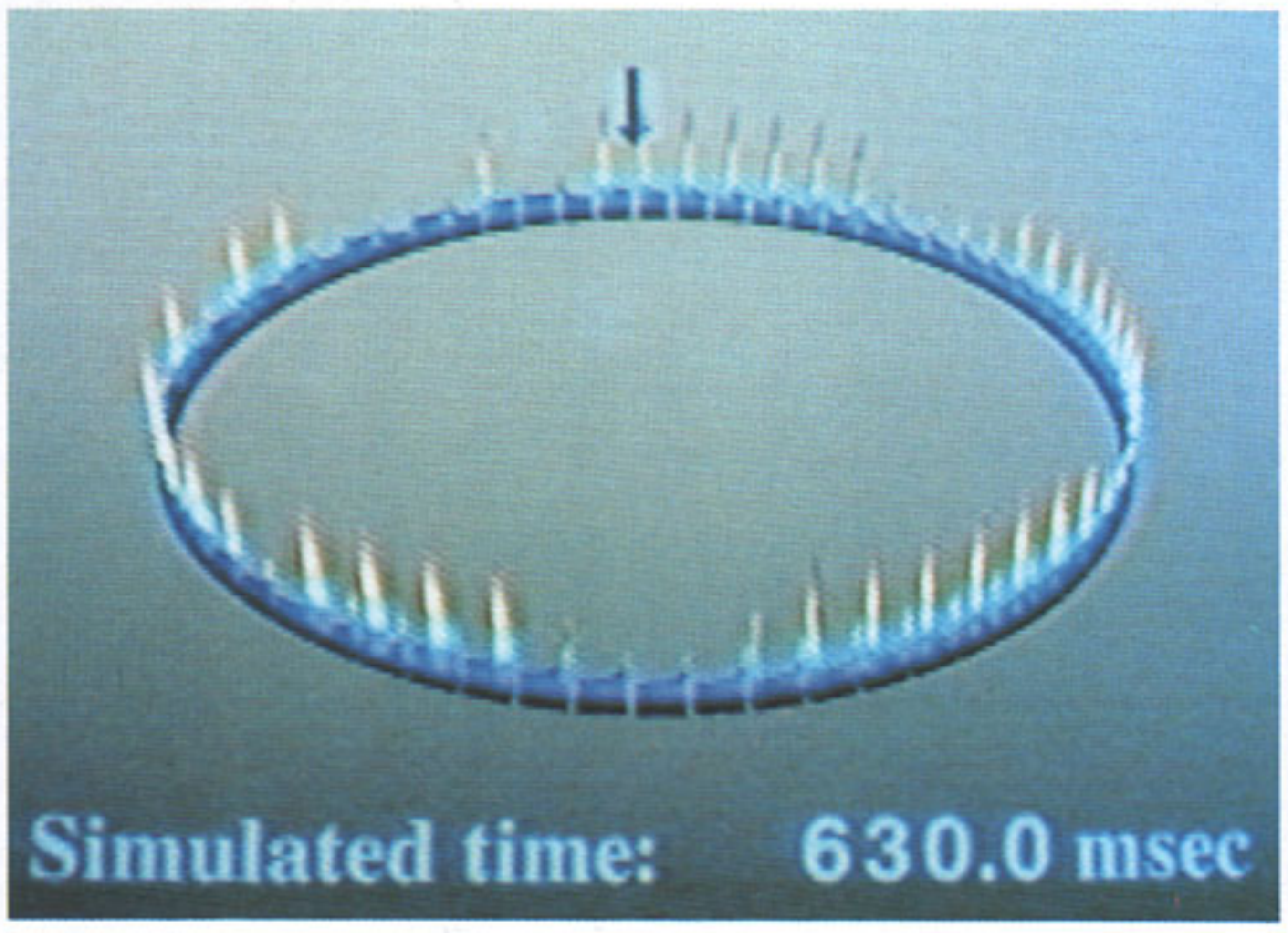



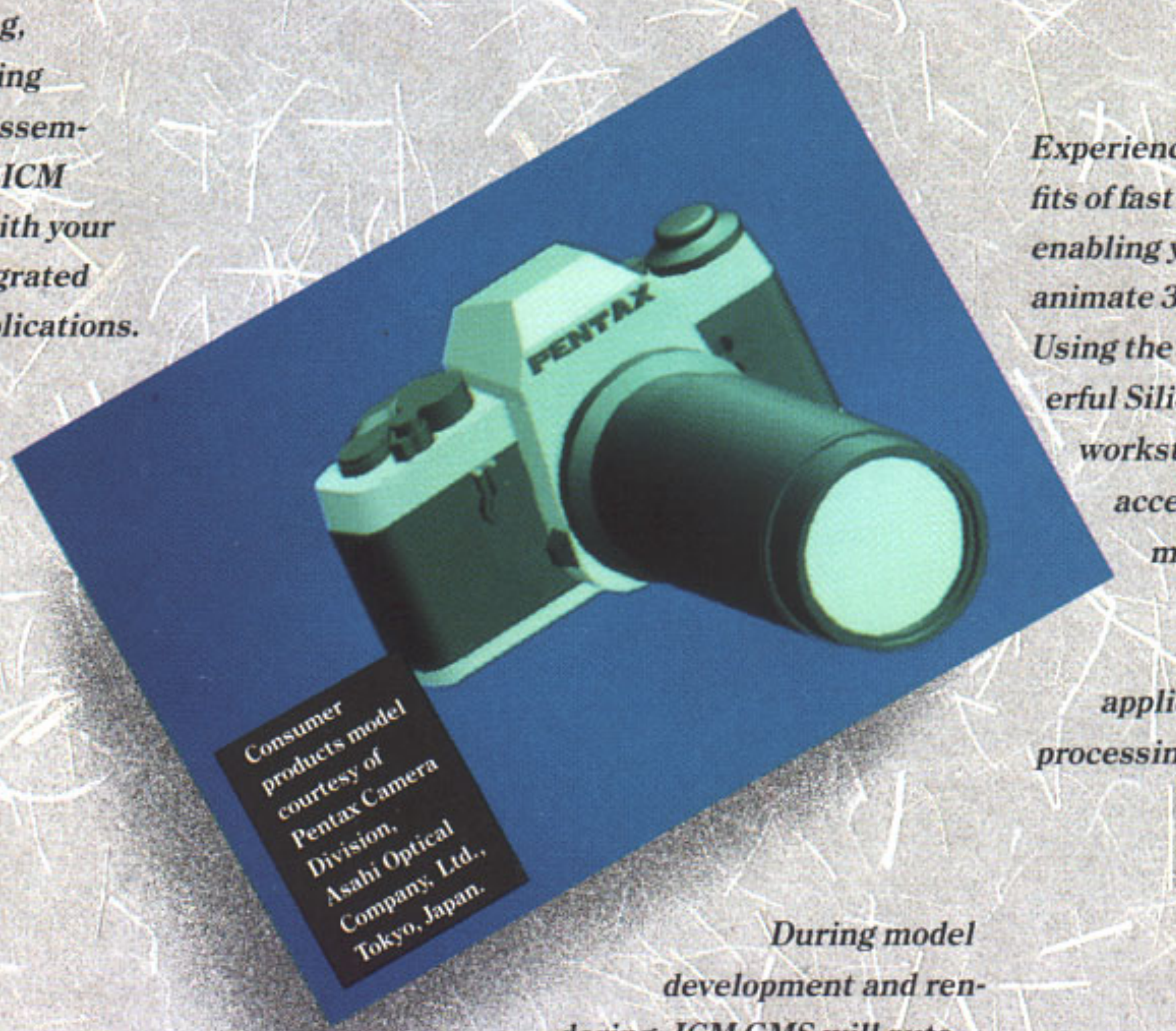

***Figure 3 and Figure 4:*** *The top image (Figure 3) visualizes the same data as that used in Figure 2. The differences include the explicit ring topology of the network and the encoding of voltage information in both a height map and colormap. This image is one frame from a videotape produced at the National Center for Supercomputing Applications (NCSA) in Urbana-Champaign, IL (produced using Wavefront Technology's rendering package on an Alliant FX/8, and then transferred to videotape). The graphical design and video production were coordinated and implemented by Stefen Fangmeier and Michelle Mercer of NCSA. The bottom image (Figure 4) is identical to Figure 3 except that a later time value in the simulation is displayed.*

pattern. Figure 4, though, shows how the interference caused by the action potentials that have already circled the ring disrupts the periodic pattern. Based on a quick viewing of our model computations, then, we might expect the long-term behavior of our system to be aperiodic.

# The Best MCAE Solution Is On Trial

*ICM GMS is the heart of your design and analysis system. It provides highly interactive tools for conceptualizing, modelling and analyzing complex parts and assemblies. You can use ICM GMS alone, or with your choice of our integrated MCAE/CAD/CAM applications.*

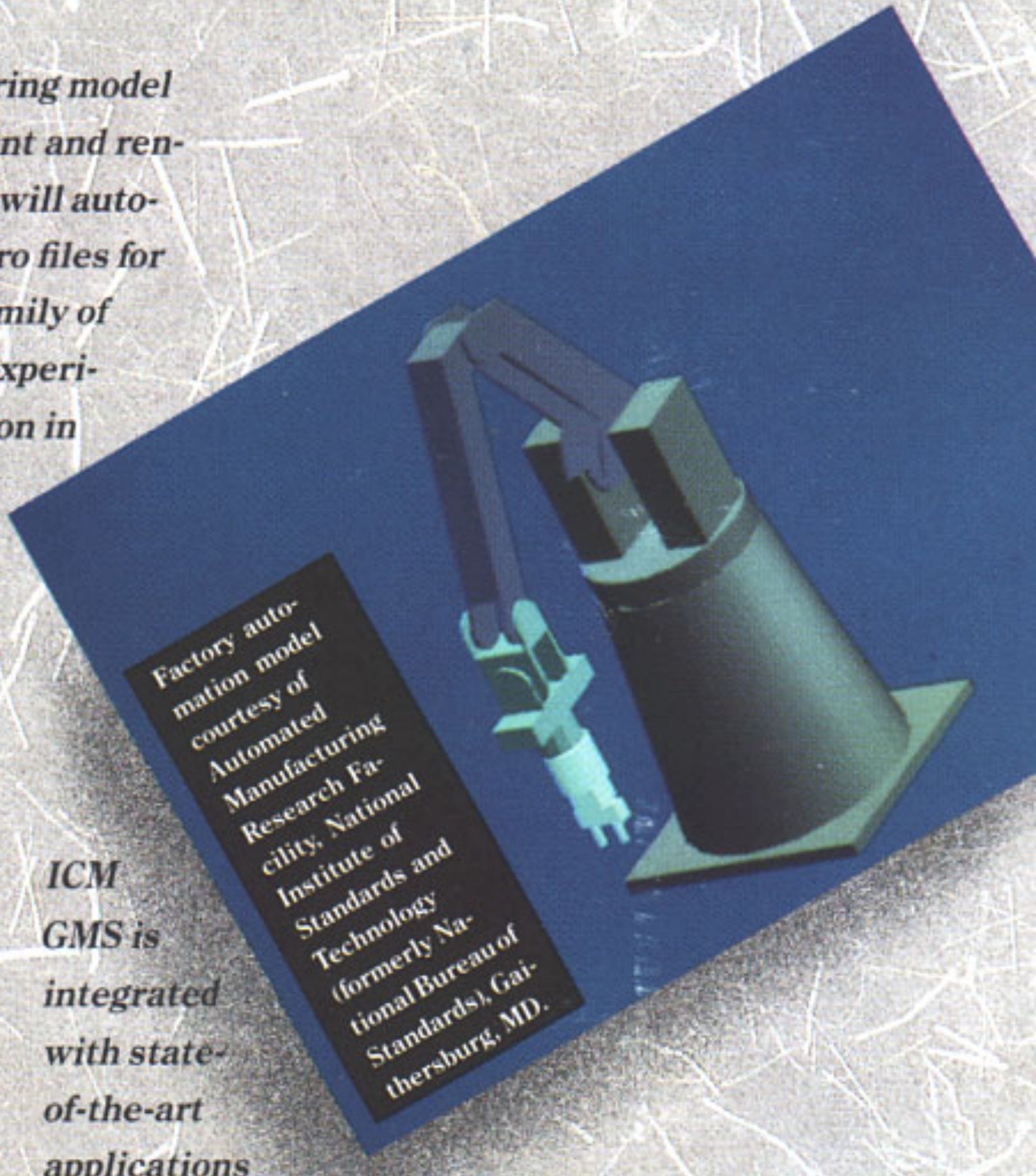
*ICM GMS gives you key tools for MCAE analysis. Operations such as testing for interferences and calculation of physical properties are direct and rapid.*




Consumer products model courtesy of Pentax Camera Division, Asahi Optical Company, Ltd., Tokyo, Japan.

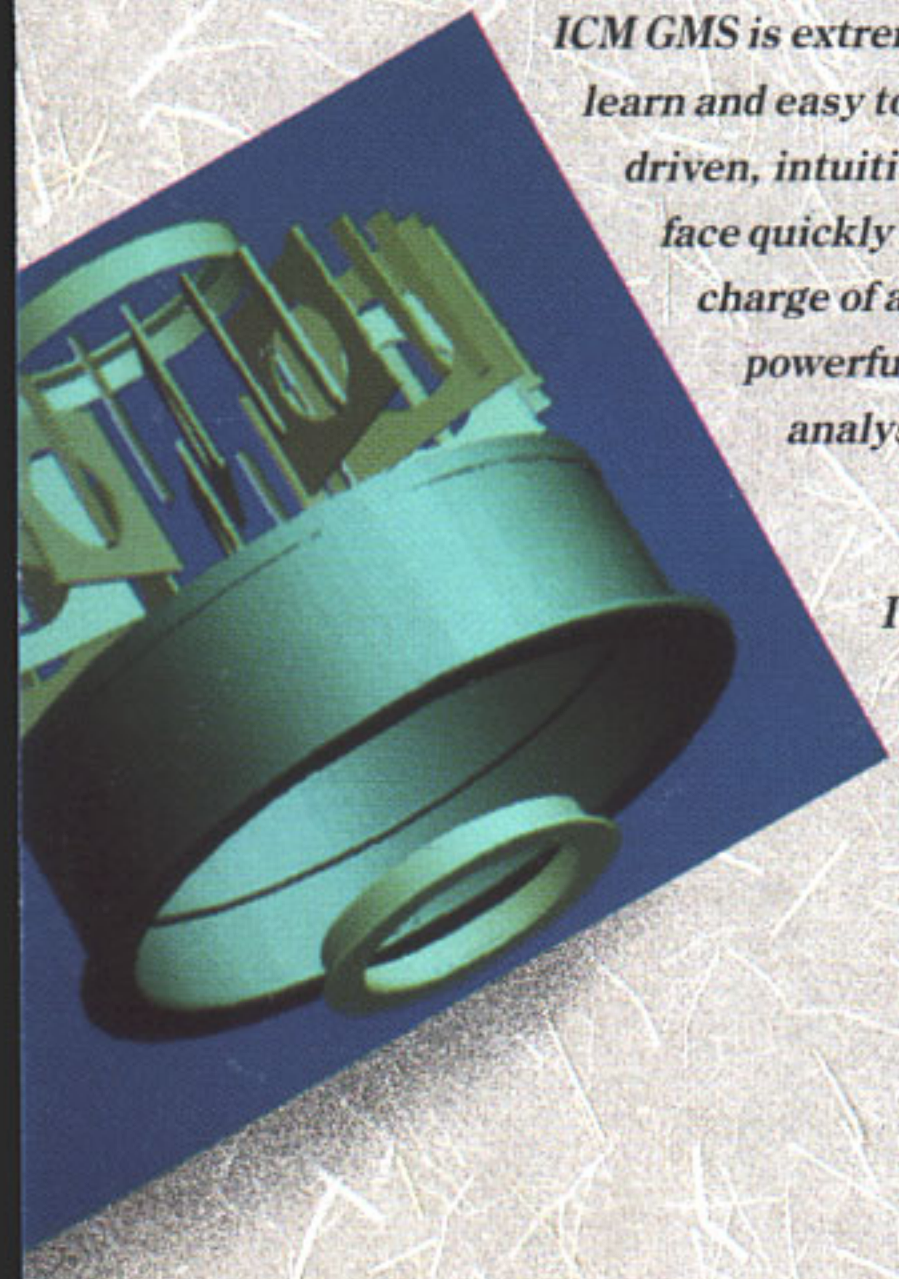
*Experience for yourself the benefits of fast ICM GMS graphics, enabling you to color-shade and animate 3D models in real time. Using the full resources of powerful Silicon Graphics IRIS workstations, ICM GMS will accelerate your complex model development. And greatly enhance your visualization of MCAE application pre- and post-processing results.*

*During model development and rendering, ICM GMS will automatically generate macro files for parametric design and family of parts generation. You'll experience measurable reduction in design cycles.*



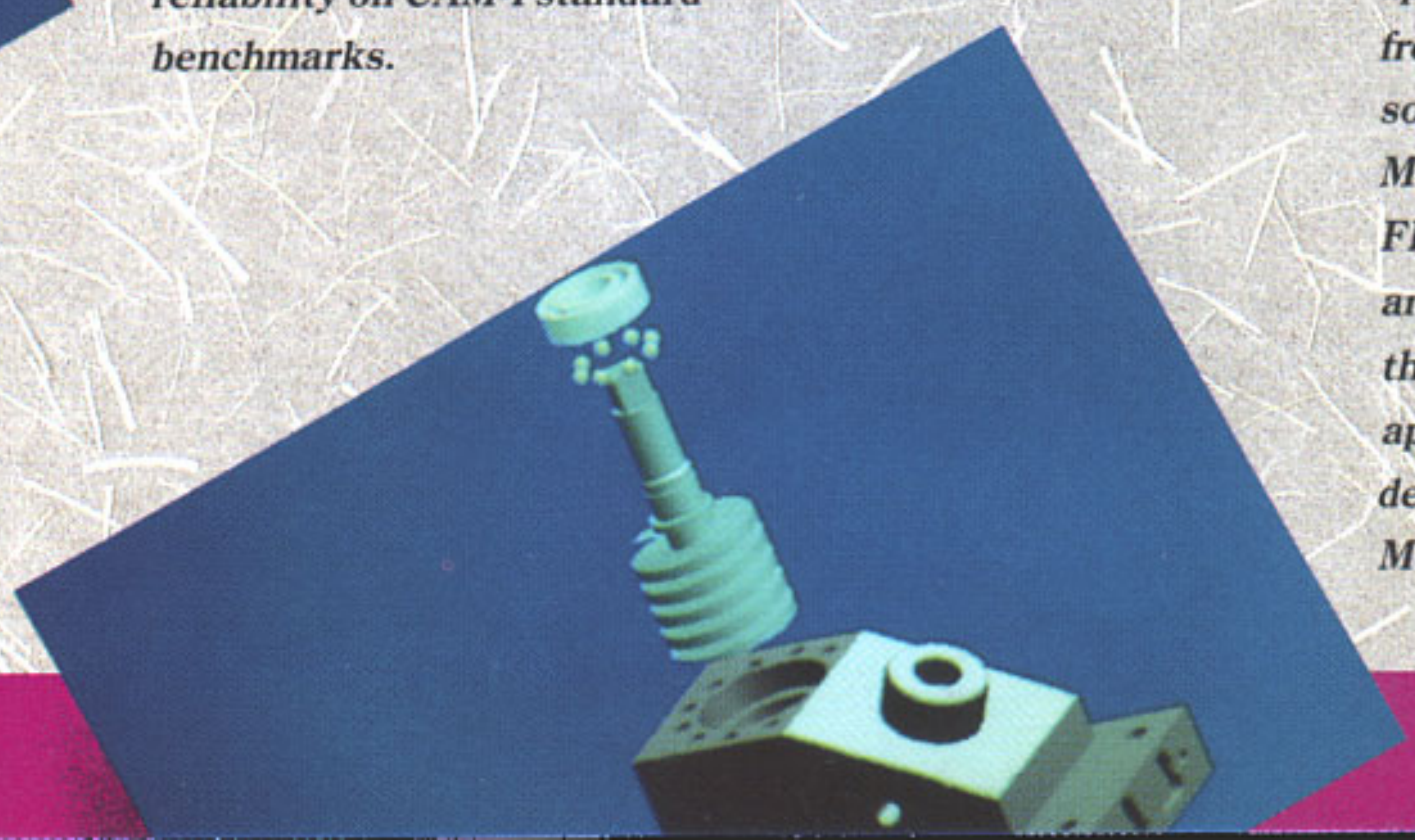
Factory automation model courtesy of Automated Manufacturing Research Facility, National Institute of Standards and Technology (formerly National Bureau of Standards), Gaithersburg, MD.

*ICM GMS is integrated with state-of-the-art applications from strategic software partners in every key MCAE area—CAD, CAM, FEM/FEA and kinematic and dynamic analysis. You get all the state-of-the-art advantages of specialized applications from leading independent developers—and the best complete MCAE solution—only from ICM.*



*ICM GMS is extremely easy to learn and easy to use. Its mouse-driven, intuitive human interface quickly puts the user in charge of a full range of powerful design and analysis tools.*

*ICM GMS performs with exceptional accuracy, efficiency and reliability on CAM-I standard benchmarks.*



# And No One's Being Charged!

Kinematic and dynamic analysis model courtesy of Automation and Robotics Laboratory, U.S. Army, ARDEC, Picatinny Arsenal, N.J.

There's no good reason to buy any MCAE software without a trial. Now you can try out the best MCAE/CAD/CAM solution in your own facilities. At no charge.

See what ICM® and its strategic software partners — today's most powerful and comprehensive MCAE solution — can do for you.

We'll train you free at ICM offices in preparation. *Then you be the judge.*

## Demand Proof

You will find that ICM is *THE LEADER* in solid modelling and visualization. You'll see first hand that ICM GMS® (ICM Geometrical Modelling System®) software provides uncompromising design and analysis tools. You'll see how ICM GMS uniquely exploits the real-time, 3D power of Silicon Graphics™ IRIS™ workstations. And you'll watch ICM GMS increase efficiency and productivity in your own facilities. Why should you commit to a system without this indisputable proof? *You shouldn't.*

## Hear The Testimony.

The next acid test in evaluating an MCAE system is its references. *Real* references. Some companies will send you a pre-screened list of users they can count on. We'll send you a *100% list of all ICM GMS customers.* An international list that includes some of the highest-powered, most experienced MCAE users, OEM's, VAR's and system integrators today. Get the unvarnished testimony of real ICM GMS customers. Consider all the evidence. *Why make a decision with less?*

## Put ICM GMS On Trial — Without Charge.

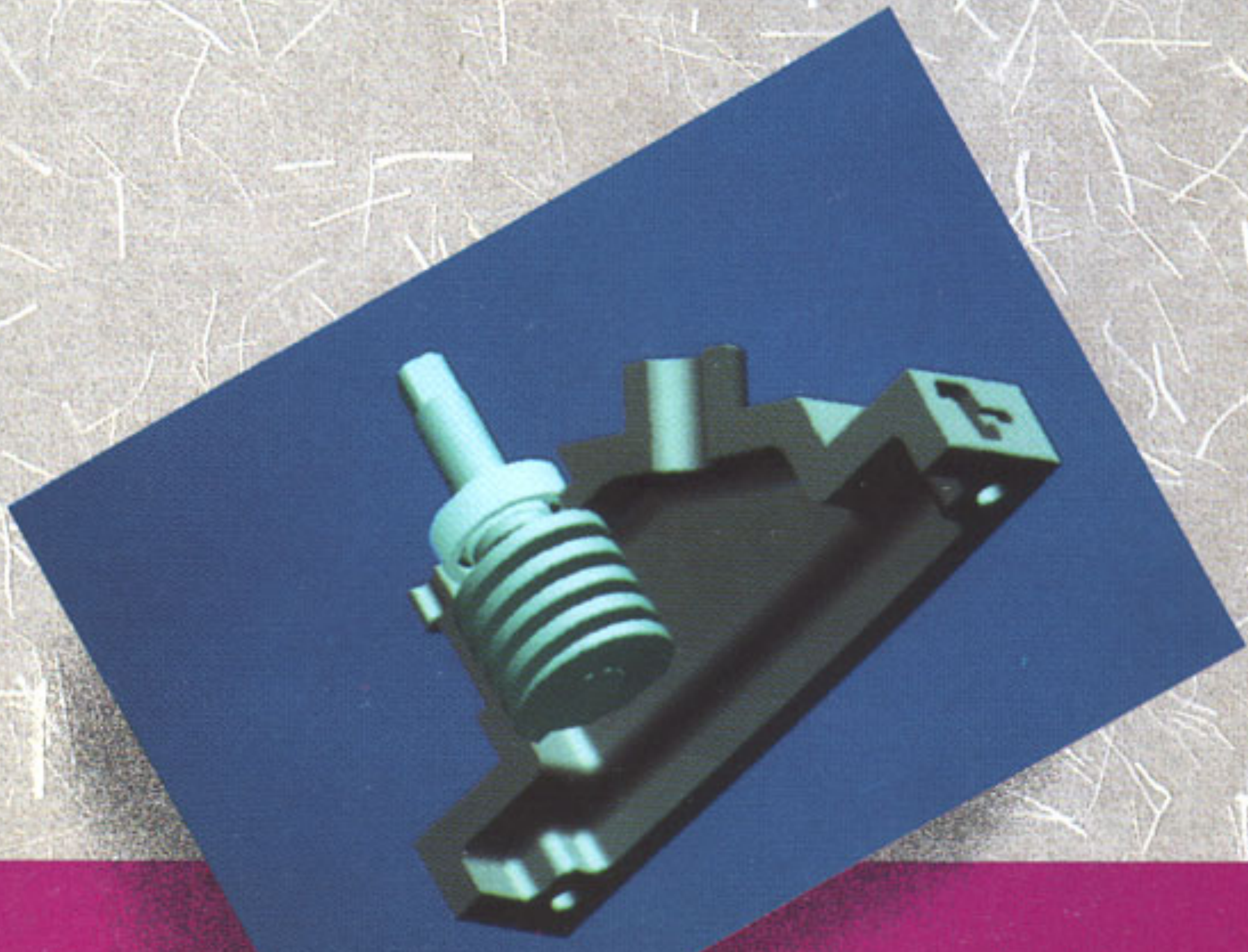
- **Arrange to try out ICM GMS.**

Don't buy any MCAE software without trial. Experience the advantages first hand. Then decide.

- **Get a 100% reference list of ICM GMS customers.**

Don't select a system without references — from all its users.

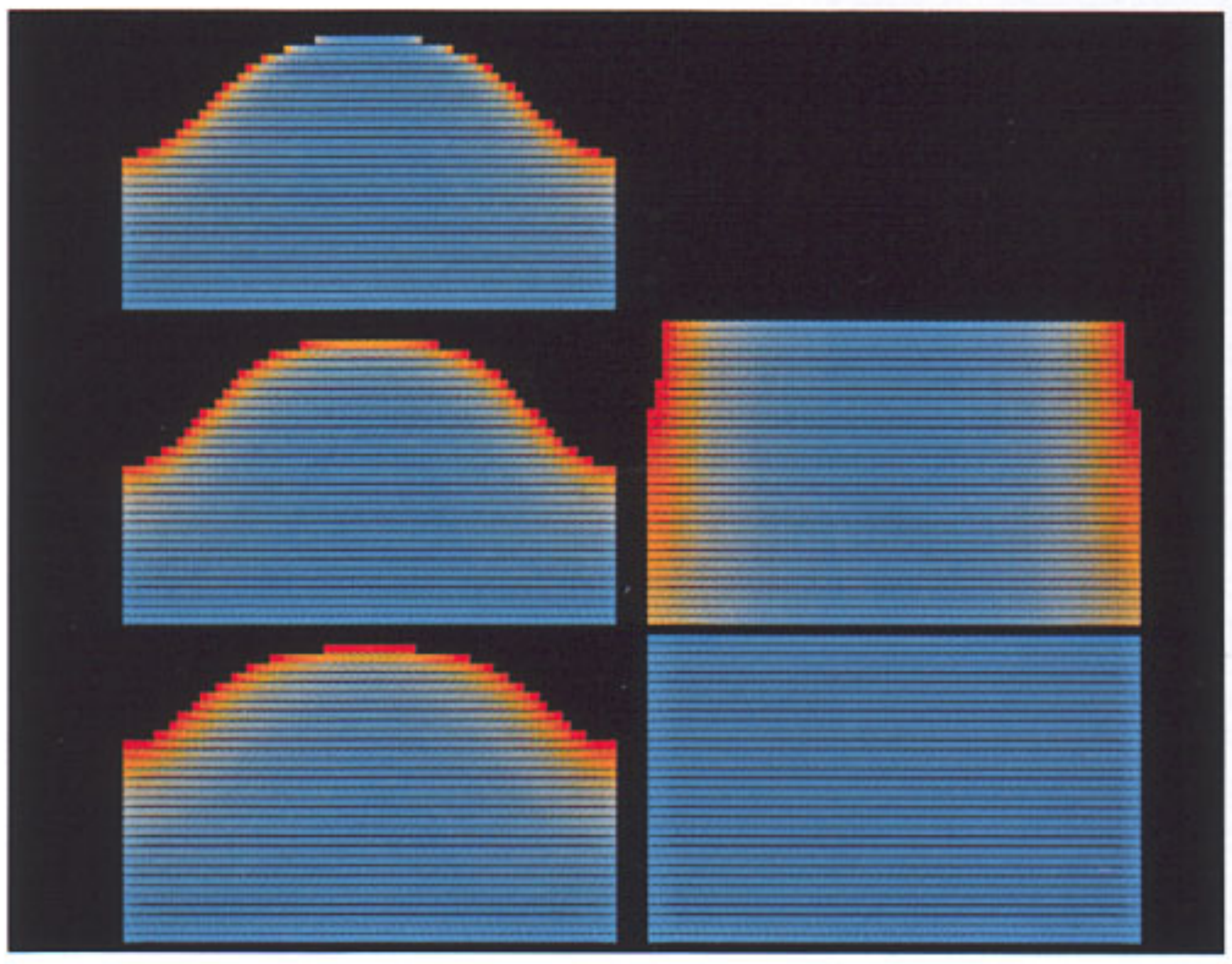
Call today! **703-476-1600**



INTERACTIVE COMPUTER MODELLING, INC.

12200 Sunrise Valley Dr. Suite 210, Reston, VA 22091

This model problem, of course, is a highly idealized caricature of the kind of neuron network one might find in living creatures. Still, it manages to retain many of the essential features defining living neurons: namely, the electrophysiological properties found in both the active and passive regions of neurons, as well as a realistic model of synaptic action. Also, the codes developed to compute solutions for this model problem can be applied much more generally than just to problems involving neurons connected end to end. These codes can also be used to simulate an arbitrarily connected ensemble of neurons with a potentially wide variety of electrical properties. In terms of modeling small networks of physiologically realistic neurons, the computational machinery developed to compute solutions to the model problem already represents a significant step toward numerically simulating small parts of the nervous system.



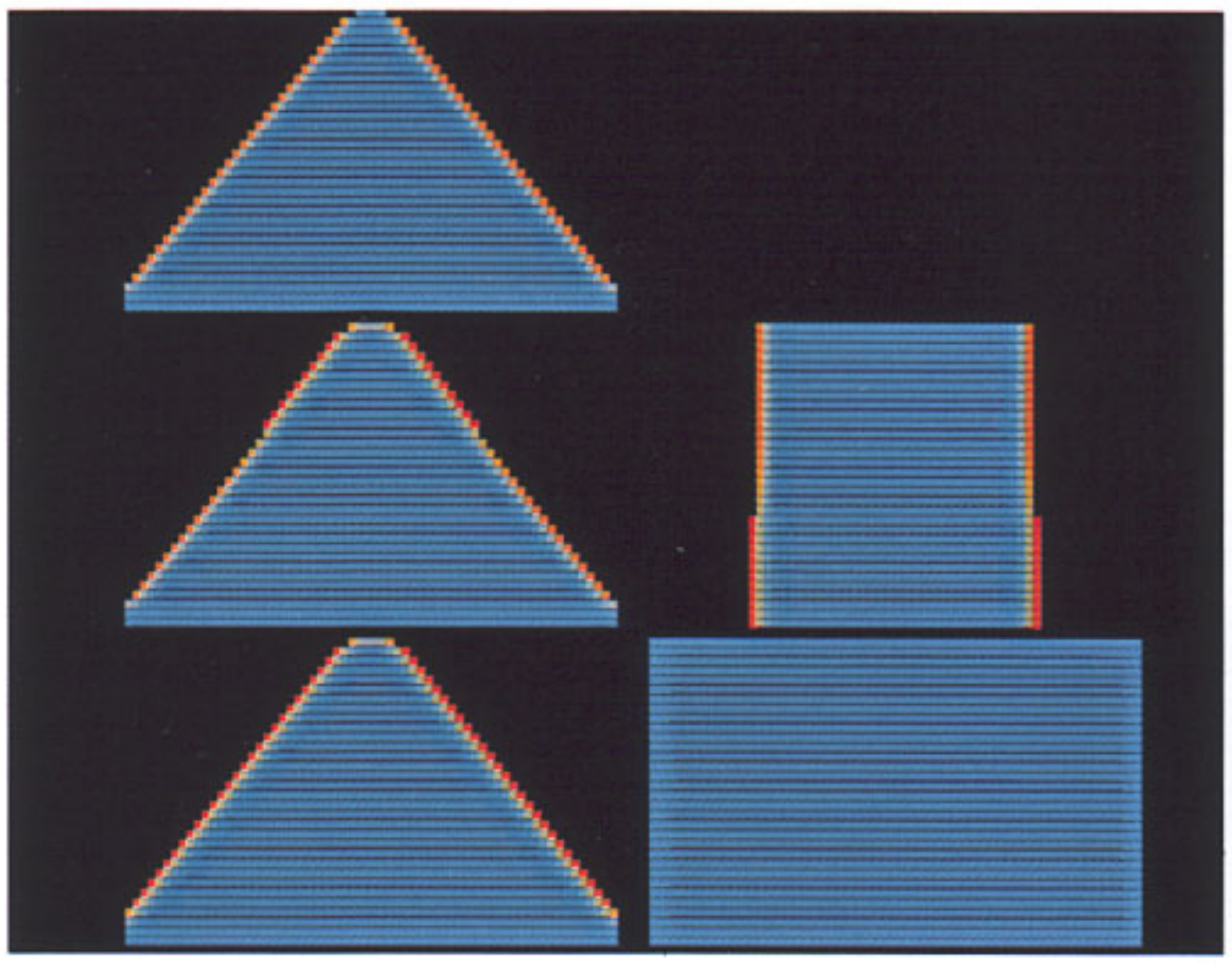


***Figure 5 and Figure 6:*** *The top image (Figure 5) shows the output of two motoneuron pools stimulated through two excitatory afferent input nerves. Each input fiber connects to all the output fibers, with the strength of the connection depending on the location of the motoneuron in the pool, and the pool to which it belongs. In this example, the strength of the connection is concentrated primarily on the closest motoneuron to the afferent fiber, diminishing rapidly for neighboring motoneurons. The ensemble of neurons simulated here include 128 afferent fibers and 128 motoneurons, 64 to a pool. The bottom image (Figure 6) is identical to Figure 5 except that the connection strength of an afferent fiber to the motoneuron pool is more diffuse. Both images were taken from the console of an IRIS 4D/60T.*

The physiology of the motor regions of the spinal cord has been one of my particular interests. The spinal cord was the first area of the mammalian central nervous system to receive extensive experimental exploration and so it remains one of the best understood areas of the central nervous system. As the starting point for modeling, then, I have devised a simple model of two cooperative motoneuron pools (a motoneuron pool is a clump of motoneurons that connect to the same functional muscle subunit) in which the two pools are excited by two afferent nerve fibers. These excitatory nerves send most of their excitation to one particular pool, but some excitation is also exerted on the other pool.

One question that might be asked is: if one varies the spatial distribution of the afferent input over the motoneuron pools, what will be the effect on the spatial and temporal pattern of motoneuron firing? Figures 5 and 6 show the results of two computational experiments done with motoneuron pools that have different values of excitatory input spatial spread. Both images, taken from an IRIS 4D/60T screen, reflect the results of three subexperiments performed with more and more excitatory fibers fired in one nerve as the number of firing fibers in the other nerve were held either at zero, half, or all fibers firing. Figure 5 shows the results with a modest amount of spread, while Figure 6 is computed with a substantial amount of spread. The areas that are colored correspond to the motoneurons that fired, with the color indicating *when* the motoneuron fired relative to the earliest firing.

The size of the temporal dispersion of the output computed in these experiments was quite unexpected -- going, in fact, against conventional wisdom, which attributes temporal dispersion solely to the variability of the con-

duction velocity of the afferent fibers. In our computations, all of the afferent fibers fired simultaneously and had identical conduction velocities.

## conclusion

The use of supercomputers in efforts to better understand the behavior of complicated mathematical models of the natural world is a general research paradigm. But, regardless of the scientific discipline in question, two of the least developed aspects of model development and refinement in a supercomputing environment are input interfaces for updating model parameters and output protocols allowing for the incorporation of computed data in many different formats with many different levels of detail. These are critical shortcomings since, in numerous disciplines, the effect on overall system behavior of events occurring on small temporal and spatial scales can be crucial to developing an adequate understanding of the system. In the nervous system, for instance, one must understand the effect of single neuron dynamics as they interact with a complicated network in order to determine what sort of cohesive output can be expected from the entire system.

It is unclear, though, what level of detail is required in describing the individual cells to capture essential network phenomenology. In fact, it presently is unclear what measures of activity are important to monitor the behavior of a network. We can be sure, however, that computer graphics will play an important role in the development of these capabilities, and that -- as the demand increases -- the role of graphical visualization will grow.

*Dr. Michael Mascagni works in the Mathematical Research Branch of the National Institutes of Diabetes, Digestive, and Kidney Diseases in Bethesda, MD, and uses the supercomputing facilities at the National Cancer Institute's Cray Center in Frederick, MD and the National Center for Supercomputing Applications at the University of Illinois/Champaign-Urbana.*

## references

[*Dodge73*] F.A. Dodge and J.W. Cooley, "Action potential of the motorneuron", *IBM Journal of Res. and Dev.*, Vol. 17, pp. 219-229 (1973).

[*Dodge79*] F.A. Dodge, "The nonuniform excitability of central neurons as exemplified by a model of the spinal motoneuron", *The Neurosciences: Fourth Study Program*, Chapter 25, pp. 583-599, The Rockefeller University Press, New York (for The Neuroscience Research Program) (1979).

[*Hodg52*] A.L. Hodgkin and A.F. Huxley, "A quantitative description of membrane current and its application to conduction in the giant axon of **Loligo**", *J. Physiol*, Vol. 117, pp. 500-544 (1952).

[*Jack75*] J.J.B. Jack, D. Noble, and R.W. Tsien, *Electric Current Flow in Excitable Cells*, Clarendon Press, Oxford (1975).

[*Masc87*] M. Mascagni, "Negative Feedback in Neural Networks", Doctoral Dissertation, Courant Institute of Mathematical Sciences, New York University (1987).

[*Masc89*] M. Mascagni, "An initial-boundary value problem of physiological significance for equations of nerve conduction", *Comm. on Pure and Appl. Math.* (at press).

[*Pesk76*] C. Peskin, "Partial Differential Equations in Biology", Courant Institute of Mathematical Sciences, Lecture Notes, New York (1976).

[*Sejn88*] T.J. Sejnowski, C. Koch, and P.S. Churchland, "Computational neuroscience", *Science*, Vol. 241, pp. 1299 (1988).

[*Traub77*] R.D. Traub, "Repetitive firing of Renshaw spinal interneurons", *Biol. Cybernetics*, Vol. 27, pp. 71-76 (1977).

# casting shadows on flat surfaces

By Thant Tessman

The ability to cast natural-looking shadows is critical to the development of realistic computer-generated images. Happily, in the case of SGI systems, the Geometry Pipeline can be used to draw an object flattened against an arbitrary plane. If the flattened object is drawn in black, it can be used as the original object's shadow. Demos such as *flight* and *insect* are examples of programs that include shadows produced in this way.

The easiest way to flatten an object is to use the **scale** function. If the ground is the plane formed by the $x$ and $z$ axes ($y$=0), then:

*scale(1.0, 0.0, 1.0)*

could be used to create a shadow from a light shining straight down the $y$ axis. Displayed in Figure 1 is a program fragment usable for producing shadows.

(By archaic convention, the $y$ axis, not the $z$, points up. To use the **lookat** function with data that assumes $z$ is up, **lookat** should be followed by **rotate (-900, 'x')** ; this will make $z$ become $y$, and $y$ become -$z$. The techniques described here can be generalized for other directions, but that is left as an exercise for the reader. )

```
zbuffer(FALSE);          /* don't bother z-buffering ground or shadow */
                         /* unless something needs to intersect it */

lookat(vx, vy, vz, px, py, pz, twist);
draw_ground();

setpattern(halftone);    /* this is so the shadow isn't solid black but */
RGBcolor(0, 0, 0);       /* only darkens every other pixel */
                         /* (assumes halftone defined already) */

pushmatrix();
scale(1.0, 0.0, 1.0);    /* shadow transformation */

position_object();       /* a set of rotates and translates that's normally */
                         /* used to position and orient the object */

draw_object_shadow();    /* the shadow version of the object should be */
                         /* the same as the object, only all black */
                         /* (no lighting or color) */

popmatrix();

setpattern(0);
zbuffer(TRUE);

position_object();       /* position and orient the object just like before, */
                         /* except this time without the shadow transform */

draw_object();           /* draw shading lit from above */
```

***Figure 1:*** *A program fragment for producing shadows.*

The **scale** function by itself is good only for shadows where the light is parallel to one of the major axes. A set of rotations and scales can be used to cast a shadow from an arbitrary direction, but it is much easier to build a single transformation matrix and use the **multmatrix** function. The hand-built matrix can then be inserted into the code fragment, replacing the **scale** function.

As shown in Figure 2, the equations for casting a shadow to the ground at $y = 0$ (assuming an infinite light source) are:

$$X_{new} = X_{old} + Y_{old}\frac{lx}{ly}$$

$$Y_{new} = 0$$

where $lx$ and $ly$ are the components of the light vector. The equation for the third dimension is:

$$Z_{new} = Z_{old} + Y_{old}\frac{lz}{ly}$$

The transformation matrix built from these equations follows:

# NEW SL-GMS (3.0)

## . . . the system you need to design dynamic models and interfaces that can display and control real-world, real-time processes . . . with speed and accuracy.

The Sherrill-Lubinski object-oriented Graphical Modeling System (SL-GMS 3.0) is an off-the-shelf software product designed to speed up your development of interactive graphics. With it you can build your own screens, displays or custom graphical interfaces.

### Helping you at the lowest level

SL-GMS is fully compatible with standard graphics packages such as GKS, CGI, IRIS-GL, X Window and SunView, among others. Normally to exploit the versatility of standard packages a great deal of code has to be written to control the graphics functions at the low level. SL-GMS eliminates this step to give users unprecedented speed and directness. Without time consuming development or study of low-level graphics functions you can easily create anything from simple screens to fully articulated dynamic visualizations.

SL-GMS handles the tasks of creating, modifying, saving, filing and displaying graphics. All calls to standard display packages are made for you by SL-GMS.

Your screens can be driven from a great variety of data streams such as external real-time sources, direct conversational input, deeper underlying models, mathematical expressions and from SL-GMS scripts for quick design evaluation and simulation.

SL-GMS supports 2D modes of operation in an integer coordinate system. It has all standard graphics primitives, including splines and bit maps, and offers hierarchical models and groups.

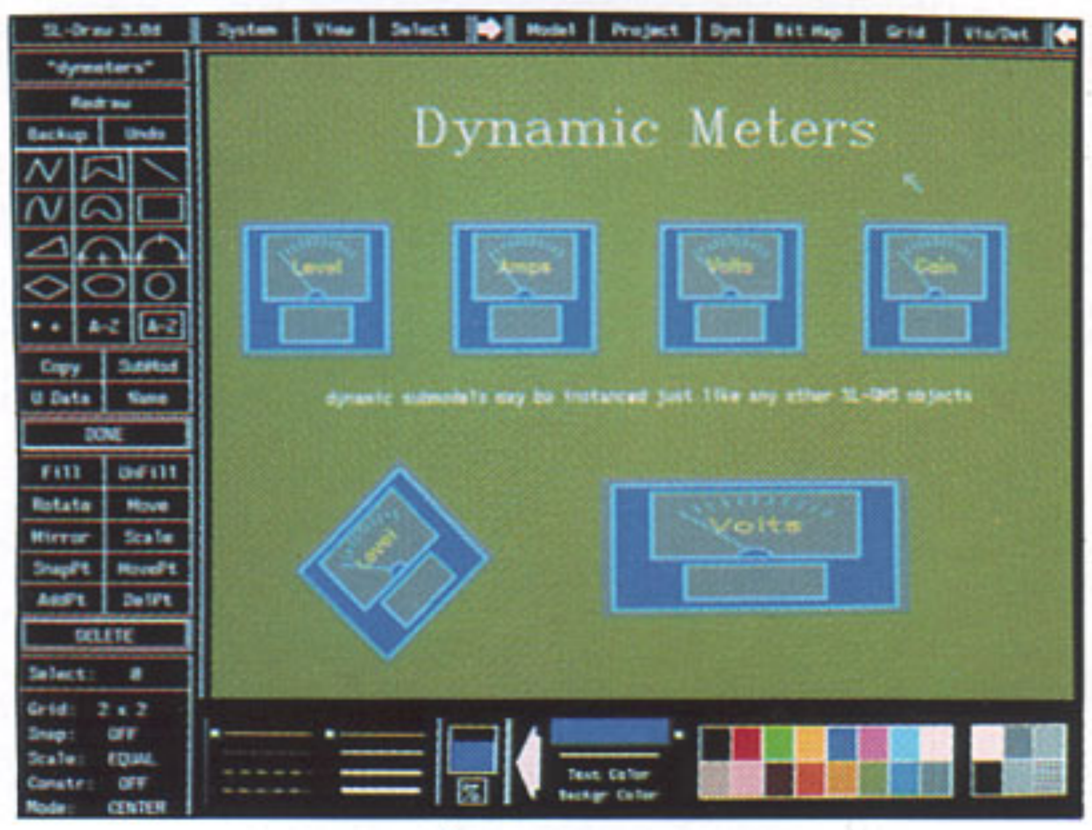
This high-level graphics management approach takes all the time and work out of handling graphical data objects. As a result you write much simpler programs to bring up screens, make changes and access objects by names you choose.

### Using SL-GMS Development tools

The **Graphical Model Editor** (SL-DRAW) is shown above, composing a process instrument-screen. SL-DRAW allows you to edit, modify, combine and file screen icons and objects which are supplied by SL-GMS—or you can easily create your own. By making appropriate menu selections or by drawing with the mouse, display screens are composed effortlessly. Attributes such as line style, width, and color may be quickly selected or changed. Objects can be moved, scaled, rotated, or mirrored. Libraries of user-defined graphics symbols may be created and referenced from many different screens.

In its dynamics mode SL-DRAW contains additional menu options which allow you to make direct associations between the iconic objects on a screen and variables in your data base. The coherent binding of icon to dynamic variable is instantaneous. The dynamic behavior of each object may be completely specified from the editor and is stored with the objects when they are saved on disk. From SL-DRAW you can also engage "GMD-run", a Preview Utility which lets you animate and evaluate your screen displays without leaving the editor. This entire construction process of your user-interface is completely codeless.

As an alternative The **Graphical Modeling Language Interpreter** (SL-GML), may be used to exercise different screen scenarios. Using the simple SL-GMS command language, screen data files may be loaded and redisplayed. Sequences of attribute or position changes may be applied to objects by name. This allows you to experiment easily with the look and feel of graphics screens before binding the screens to your real-time program variables.




### SL-GMS Run-Time Module

While written in C, SL-GMS is uniquely hospitable to other languages and code. It is extremely adaptable and its run-time module configured by development tools, is easily embedded, in part or in whole, in a user's application program. This module consists of elements from the **Graphical Modeling Function Library** (SL-GMF), which consists of all the functions needed to impart and alter the attributes of objects or icons you have created with SL-DRAW. The **Graphical Model Dynamics Library** (SL-GMD) contains a powerful set of procedures and functions to support the codeless binding of icons as they are created with SL-DRAW directly to application variables. The automatic invocation of these functions permits the change of any or all icon attributes, e.g., shape, appearance, size, color, position, orientation, visibility.

SL-GMD also allows formulas and expressions of variable combinations to control icons and displays.

Creating graphics display screens in this way eliminates the work necessary to attach even complex graphics to your applications. This is especially true if your requirement must maintain compatibility with graphics standards.

The system contains a separate object layer called SL-GWS—Graphics WorkStation function library—to ensure and simplify the transportability of SL-GMS to the widest variety of systems and devices.

### SL-GMS Architecture

SL-GMS is a solid and robust environment. It is a true, object-oriented, integrated, coherent, hierarchical-modeling system with all its components—SL-DRAW, for example—built up from combinations of its own graphics primitives and from functions in the SL-GMF library. These functions are organized according to classes and subclasses.

SL-GMS has particular value to many developers because of its strict underlying architecture, managed and controlled by the Object-Oriented Environment (SL-OOE). This is itself a library of versatile object-management functions, used within SL-GMS, and which is made accessible to interested users for application program development. SL-OOE also contains a program utility to define the user structural-framework and organize program code into strict hierarchical classifications. SL-GMS uses run-time class-information to provide the functionality present in object-oriented programming systems, such as messages to objects, dynamic binding, encapsulation, class inheritance, and so on. SL-OOE does not require the use of any language extensions or any special compilers.

Useful object management functions include: 1) an automatic facility for debugging and tracing function calls invoked on every object, 2) automatic archival and retrieval of arbitrary data structures, and 3) automatic "dumping" of object data.

SL-OOE together with (SL-GMF) can be seen as functionally similar to the emerging Programming Hierarchical Interface Graphics Standard, PHIGS. However, SL-GMS is much more extensible than PHIGS with the same functionality, and SL-GMS contains the editor SL-DRAW and the coherent-binding capability. But SL-GMS can easily interface to PHIGS through the SL-GWS layer for users with this requirement.

Complete SL-DRAW source-code is offered at a low price, so that you may customize the editor to your particular requirements. For example, you may want to change the layout of SL-DRAW menus, or offer a different set of attributes or primitives to the user.

### License Prices and availability.

| | | |
|---|---|---|
| **Development Configurations** | | Components |
| SL-GMS | $ 9,500 | (SL-DRAW/GML/GMF) |
| SL-GMS-D | 12,500 | (above, plus SL-GMD) |
| **Run-Time Licenses (Optional)** | | |
| SL-GMS | $ 2,400 | (SL-GMF) |
| SL-GMS-D | 3,600 | (SL-GMF plus GMD) |
| **Source code (Optional)** | | |
| SL-DRAW | $ 9,500 | |
| SL-GML/GMF | 55,000 | |
| SL-GMD | 20,000 | |

Available: UNIXbsd 4.2, 4.3, System V, Ultrix, DEC-VMS
GKS level 2c (DEC Version 3.0 and others)
CGI, SunView, IRIS GL, X Window, DECwindows, Silicon Graphics IRIS 3130 & 4D/70, SUN3, SUN4, VAX, MicroVAX

**Call or write today.**


SL

**Sherrill-Lubinski**
Suite 110 Hunt Plaza
240 Tamal Vista Boulevard
Corte Madera, California 94925
415/927-1724

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{lx}{ly} & 0 & \frac{lz}{ly} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

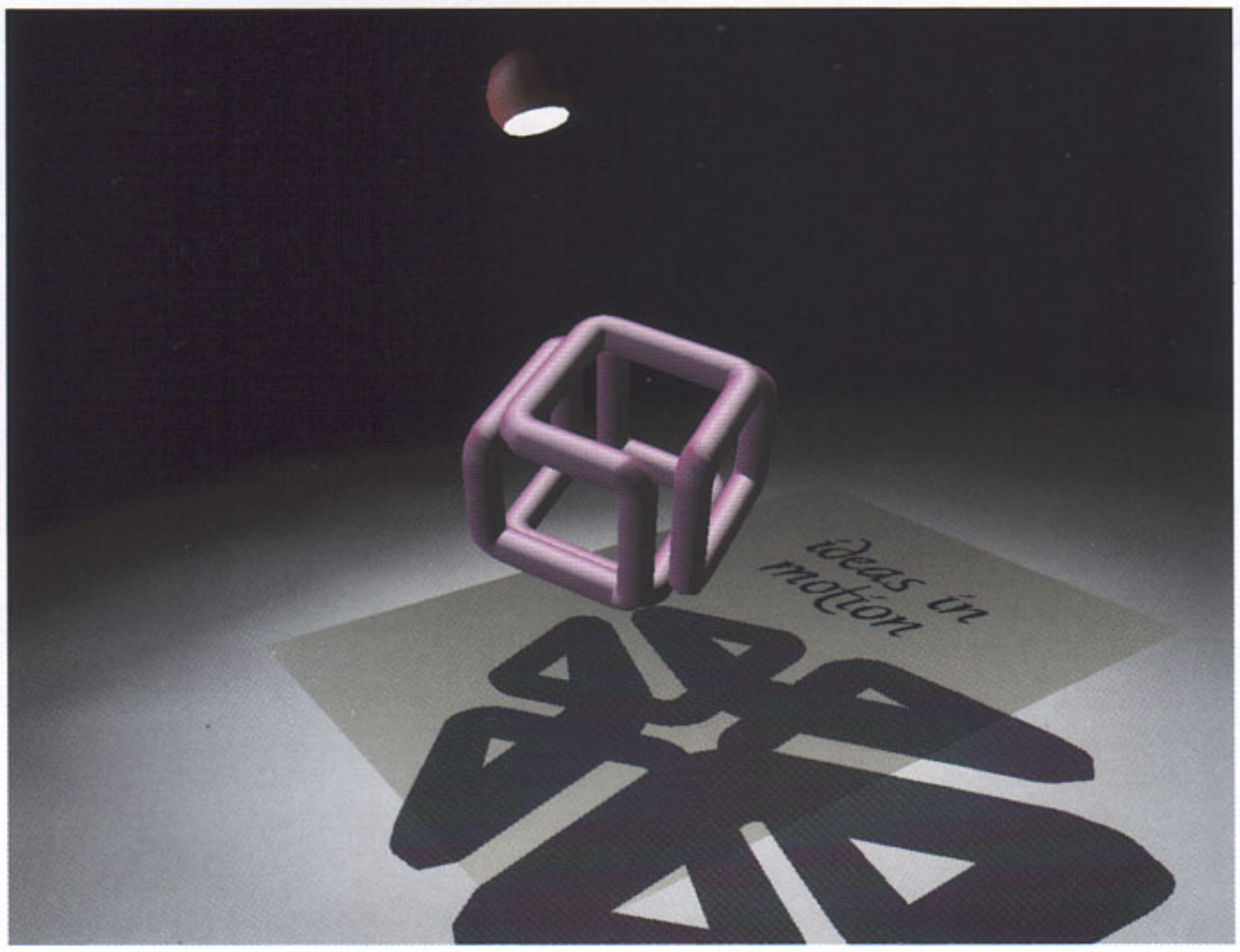
Note that if *lx* and *lz* are zero, (*i.e.* if the light is shining straight down), the matrix will do the same thing **scale(1.0, 0.0, 1.0)** would.

This matrix and the one created by the **scale** function produce shadows as if the light were infinite. In this sense, they are orthogonal. A matrix very similar to the one created by the **perspective** function will generate shadows from a local light source (see Figure 3).

Here, *lx* and *ly* are the absolute coordinates of the light, because the position of the point relative to the light must be used. Just as before, the ground is at $y=0$:

$$Y_{new} = 0$$

By similar triangles:




*A still from "Ideas in Motion: The Movie", written in C and generated in real time (approximately eight to fourteen frames per second, depending on window size) using an IRIS 4D/70GT.*

$$\frac{lx - X_{new}}{ly} = \frac{lx - X_{old}}{ly - Y_{old}}$$

where $ly$ really is $ly - Y_{new}$ but $Y_{new}$ is zero.

The fourth column in transformation matrices (the $w$ column) is used for the perspective divide. A matrix really transforms $x$, $y$, $z$, and $w$, where $w$ usually is 1.

The resulting $x$, $y$, $z$, and $w$ are normalized to make $w=1$ (by dividing

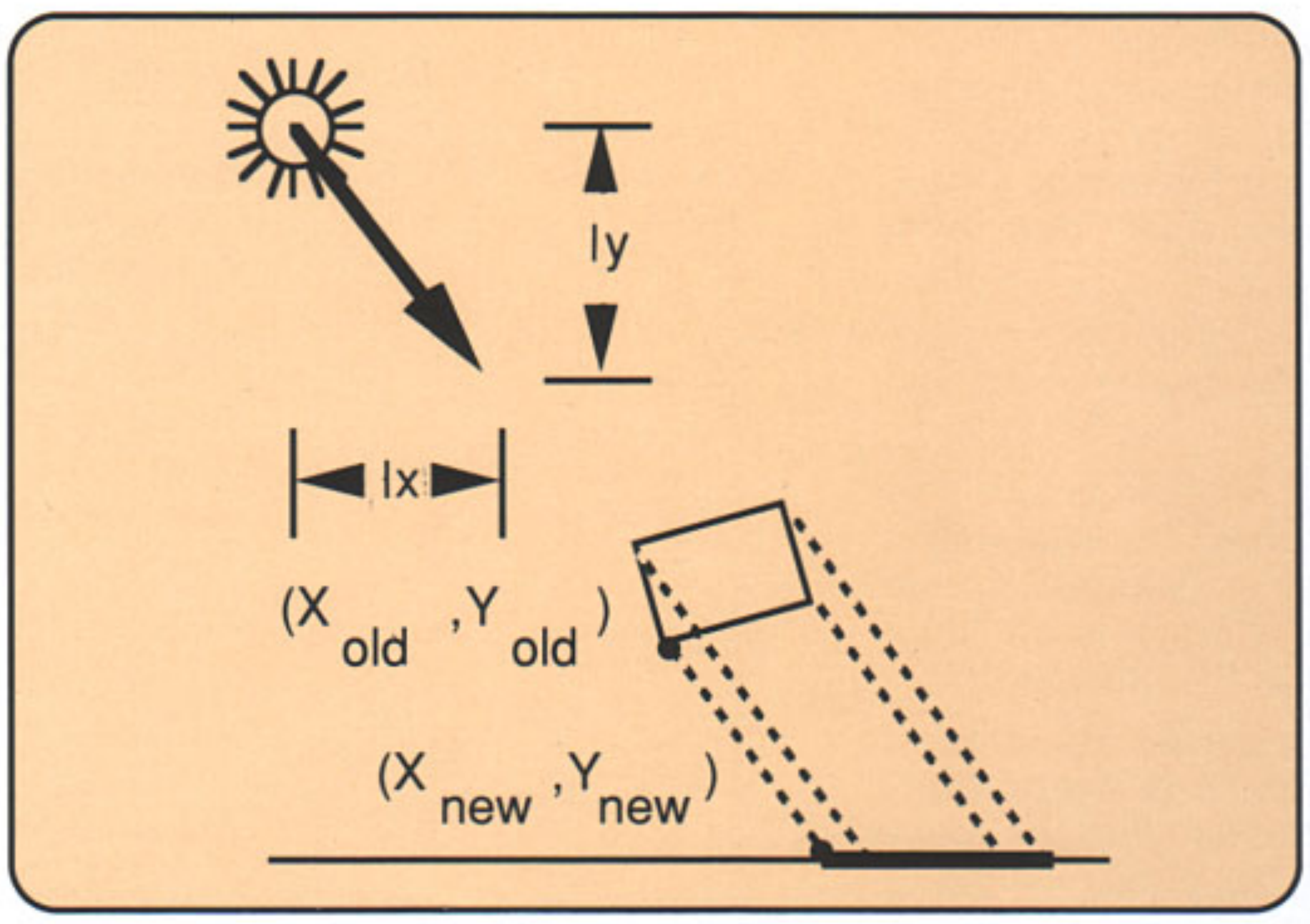



***Figure 2:*** *Shadow from an infinite light source.*

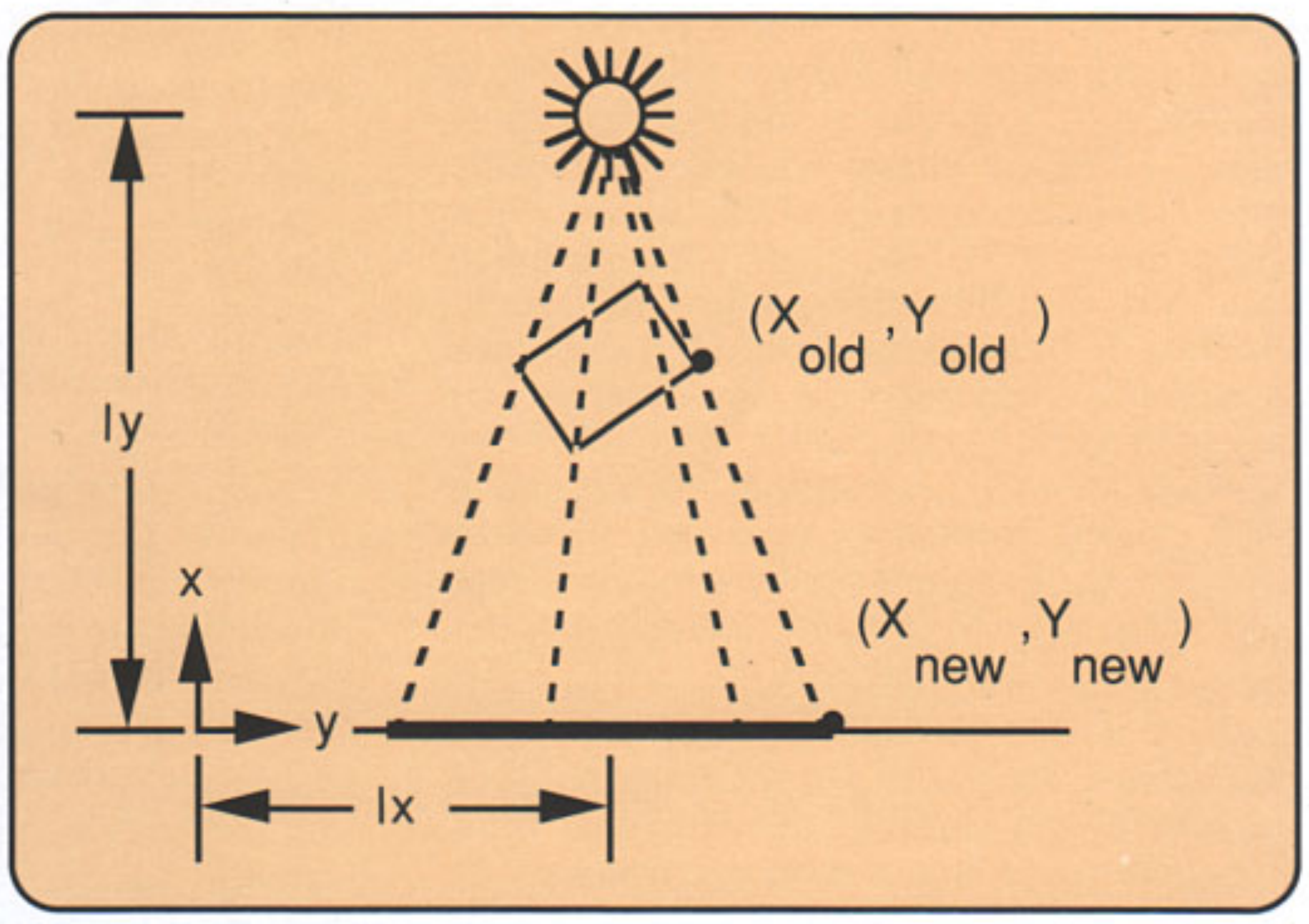



***Figure 3:*** *Shadow from a local light source.*

each of them by $w$). Solving for $X_{new}$:

$$X_{new} = \frac{ly\, X_{old} - lx\, Y_{old}}{ly - Y_{old}}$$

For constructing the matrix, $W_{new}$ is $ly - Y_{old}$. Similarly for $Z_{new}$:

$$Z_{new} = \frac{ly\, Z_{old} - lz\, Y_{old}}{ly - Y_{old}}$$

The transformation matrix built from these equations is:

$$\begin{bmatrix} ly & 0 & 0 & 0 \\ -lx & 0 & -lz & -1 \\ 0 & 0 & ly & 0 \\ 0 & 0 & 0 & ly \end{bmatrix}$$

Although this technique is very specialized, it also is very quick. In fact, the overhead is low enough that there is no reason *not* to rebuild a matrix for each frame to allow for a moving light source. Combined with hardware-assisted lighting, the dynamic images that result can look surprisingly realistic.

*Thant Tessman is a Silicon Graphics systems engineer working in the company's Advanced Systems Division.*

*For more information on transformation matrices, see Mathematical Elements for Computer Graphics by David F. Rogers and J. Alan Adams (McGraw-Hill, 1976). Also see Chapter 4, "Coordinate Transformations", and Appendix C: "Transformation Matrices" in the Graphics Library User's Guide, Volumes I and II.*

## SILICON GRAPHICS, INC. EDUCATION CENTER COURSE CALENDAR

**(Through June 1989)**

| COURSE | LOCATION* WEC | EEC |
|---|---|---|
| 3000 SERIES COURSES | WEC | EEC |
| GRAPHICS I<br>4.5 days | Feb 27, 1989<br>Jun 26, 1989 | not available |
| SYSTEM ADMINISTRATION<br>4.5 days | Apr 24, 1989 | not available |
| SYSTEM MAINTENANCE<br>10.0 days | May 15, 1989 | not available |
| 4D SERIES COURSES | | |
| GRAPHICS I<br>4.5 days | Mar 13, 1989<br>May 1, 1989<br>Jun 12, 1989 | Feb 27, 1989<br>Apr 10, 1989<br>Jun 5,1989 |
| ADVANCED GRAPHICS<br>3.5 days | Mar 20, 1989<br>Jun 19, 1989 | Apr 17, 1989 |
| PARALLEL PROGRAMMING | Mar 27, 1989 | May 22, 1989 |
| SYSTEM ACCELERATOR<br>4.5 days | Apr 3, 1989<br>May 8, 1989 | Mar 6, 1989<br>May 1, 1989<br>Jun 12, 1989 |
| SYSTEM ADMINISTRATION<br>4.5 days | May 15, 1989 | Mar 13, 1989<br>Jun 19, 1989 |
| NETWORK ADMINISTRATION<br>4.5 days | Mar 6, 1989<br>Jun 5, 1989 | Apr 3, 1989 |
| SYSTEM MAINTENANCE<br>10.0 days | Apr 10, 1989 | May 8, 1989 |
| PERSONAL IRIS SYSTEM MAINTENANCE<br>3.5 days | Mar 27, 1989<br>May 8, 1989 | not available |

KEY:
WEC--*SGI Western Education Center, Mountain View, CA*
EEC--*Eastern Education Center, SGI Federal, Bethesda, MD*

*The SGI Education Center reserves the right to cancel classes due to insufficient enrollment.

# quelling network nightmares

By Michael Smith

Efficient administration of a single workstation can be a formidable task, given that the system administrator must:

- assure that each account on the workstation is maintained and password protected;
- assure that users are categorized into network-wide groupings so as to implement file system territorial protection;
- ensure the availability of networking services;
- maintain a list of host names and addresses consistent with the lists kept by other systems on the network;
- ensure that the system's set of configuration files are up-to-date.

Now, repeat these tasks across a 20-system network responsible for servicing 80 user accounts and supporting network resource sharing through **rcp, rsh,** and **rlogin**, and spread the work over a square kilometer or so. Sound intimidating? It doesn't have to be. Knowledge and an understanding of the right networking tools for the job can make a computing environment of this sort work like a dream.

The implementation of network administration tools such as the Yellow Pages (YP®) Database Service can significantly ease the administration burden for any particular network. The Yellow Pages is a network facility that centralizes the network's various administrative activities on a single workstation, known as the *YP Server.* Responsible for maintaining a master copy of various network and workstation configuration files, the YP Server is where all network-wide changes are made and propagated to each remote host. As a result, a consistent set of configuration files is available to each local host, or "Client", using the YP services.

Each file is compiled into a database format, called a *YP Map,* which can be quickly accessed to efficiently process the requests made by network hosts.

## how it works

The Master Server maintains the file contents for the master network configuration in its YP maps. This way, YP clients, instead of accessing their local configuration files for requests, can be directed to the Master Server's map.

The Master Server listens for requests from YP clients and, upon receipt, queries the appropriate YP map for the information the Client requires. Once the information has been served, the operation can finally be completed locally by the Client.

The YP Master Server can also maintain one or more backup maps residing on hosts called *Slave Servers,* which can be used, when needed, as temporary Master Servers. However, changes to the network configuration files can be made only on the Master.

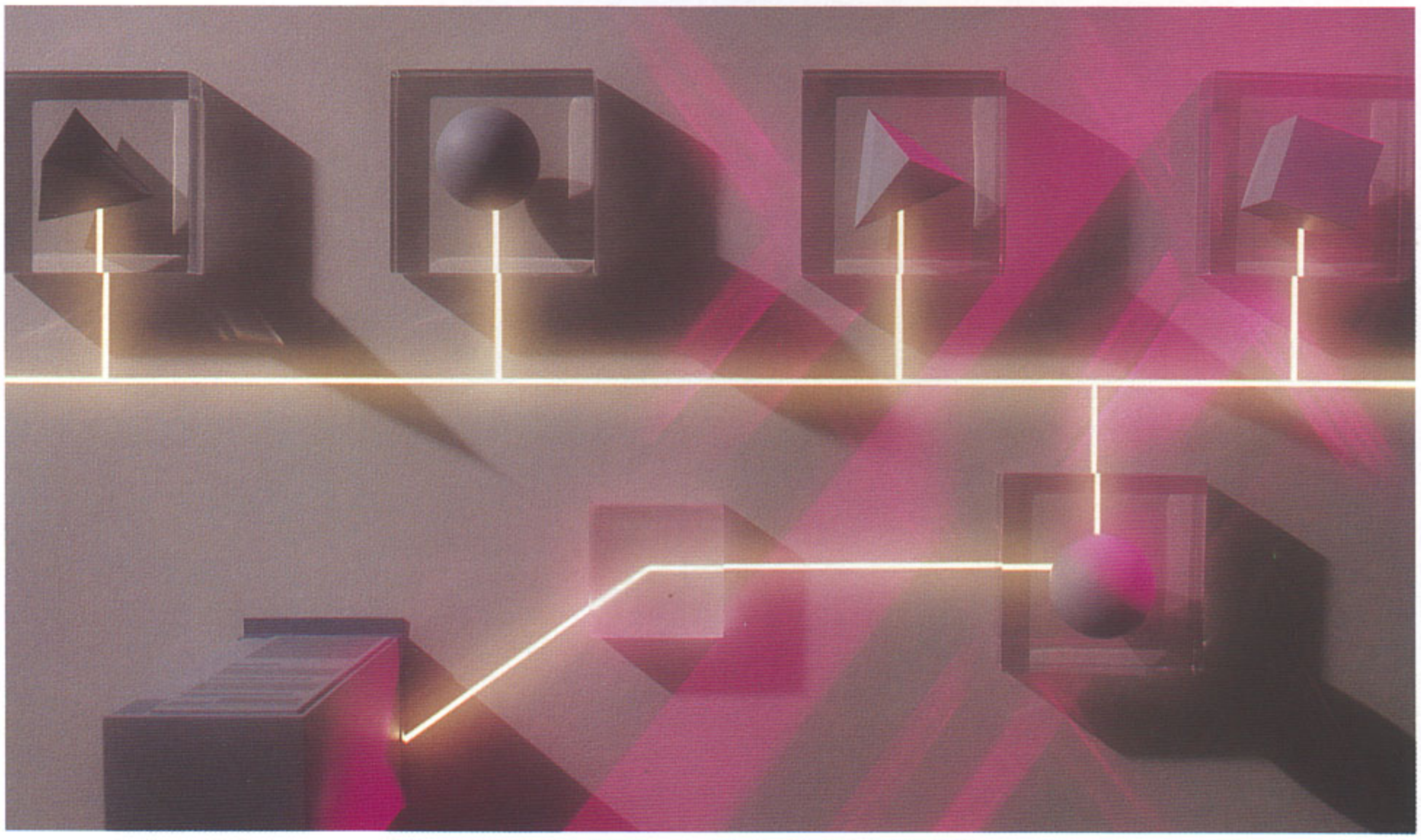
## setting up YP

Before initializing YP, one workstation must be designated as Master Server, responsible for serving global configuration information from its YP maps. It is important that this be a reliable machine since it contains essential network information. The burden on Slave Servers is not nearly so great, but they too should be fairly stable. All remaining workstations are candidates to become YP Clients.

A "domain" is defined as a set of entities (or workstations) served by a single administrative body; its name should conform to the domain-naming convention for your associated internet. Of course, should your network not be directly connected to an internetwork supporting the domain architecture, such as the ARPA Internet, you're free to select any name you wish. For example:

```
doc.sgi.com
```

could be used to refer to the host *doc* in the domain *sgi.com.*



# ARE YOU A SLAVE TO YOUR NETWORK?

Network Training Makes You The Master

## Imagine Your Network Working For You

At Silicon Graphics, we believe your network should make your computing environment more powerful and your job a lot easier. We not only offer you the best networking products in the industry but also the training that enables you to take full advantage of these products. The IRIS **4D NETWORK ADMINISTRATION** course will teach you to master Silicon Graphics' data communication tools and networking services. But mostly you will learn how to get your network up and running fast and efficiently.

## Training Makes Your Network Hummm

Our **NETWORK ADMINISTRATION** course provides in depth coverage of networking products and tools including:

- **UUCP**
- **Ethernet and TCP/IP**
- **Telnet and FTP**
- **NFS**
- **Yellow Pages**
- **Security**

You will gain the skills and knowledge that you need to create the network of your dreams.

**Connect with Customer Education. Call for a course schedule today.**
**(800) 356-9492**

SiliconGraphics
Computer Systems

Before installation, a file called *ypdomain* must be created in the YP directory */usr/etc/yp*. The file, which contains your domain name, must exist on all designated YP servers, and Clients must also have the file.

## initializing the YP maps

Before any Slave Servers and Clients can be configured, the Master Server must be configured and running. To proceed with the installation, the configuration files to be included as maps should be brought up-to-date and backups ought to be made. Next, the initialization process for YP maps, **ypinit**, can be executed as follows:

```
ypinit -m
```

Once this is done, **ypinit** builds a map called *ypservers* from input provided at the command line. At this point, enter the selected host names of the Slave Servers. When **ypinit** has finished, the daemons can be started from the command line. The following Master Server daemons must also be started:

```
/usr/etc/ypserv
/usr/etc/ypbind
/usr/etc/rpc.passwd
```

Now, the Slave Servers can be initialized. At each server-to-be, the command:

```
ypinit -s doc
```

initializes the Slave Server's YP maps by retrieving those created on the host *doc*, the Master Server. Once this is done, the Slave Server(s) will contain identical copies. As before, manually start the YP daemons whenever **ypinit** completes, like so:

```
/usr/etc/ypserv
/usr/etc/ypbind
```

To start up your YP Clients, initialize only the daemon:

```
/usr/etc/ypbind
```

## firing up the daemons

In IRIX version 3.1, provisions are made to enable or disable various system options from the command line. Prior releases required that you edit configuration files directly from */etc/init.d*, either adding or deleting comments from file entries. In release 3.1, the **chkconfig** command simplifies the process. The commands:

```
/etc/chkconfig ypmaster on
/etc/chkconfig ypserv on
/etc/chkconfig yp on
```

will start up the appropriate YP servers for the Master Server, Slave Server, and Client, respectively.

## map propagation

As the network evolves, configuration files will be modified to add new users, new netgroups, and new host addresses, as well as to remove obsolete users and host addresses. Thus, to maintain consistency, not to mention accuracy, the Master Server's maps must be periodically propagated. This propagation can be performed manually, or automatically, using the YP utility, **ypxfr**, which can be executed from the command line. For example:

```
/usr/etc/ypxfr hosts.byname
```

transfers the YP map *hosts.byname* to all Slave Servers.

To ensure that YP maps are propagated periodically, the **ypxfr** execution can be kicked off at set intervals by **cron**. Three example scripts in */usr/etx/yp* are:

```
ypxfr_1perhour
ypxfr_1perday
ypxfr_2perday
```

These files contain individual **ypxfr** entries for each map to be propagated and may be altered to ensure that all are included. You can also be selective about the propagation interval by having the more active maps, like the one related to the password file, be propagated more often. The most important thing is that information across the network be kept consistent.

## customizing YP clients

In order to direct a YP Client to use the YP Master password map exclusively for user logins, the Client can

add special characters to various local configuration files. These special characters actually force the Client to access the Master Server YP map corresponding to the Client's file.

Clients can also include or exclude specific user, group, and netgroup access to their workstations.

## the password file

Normally, when users or processes on a Client need access to a configuration file, they'll use the one that's local. A "+" character, though, can be used to force the use of the Master Server's YP map. In the password file containing the following information:

```
lp: :9:9:00000-lp (0000): /usr/spool/lp:
nuucp::10:10:0000-
uucp(0000):/usr/spool/uucppublic:/usr/lib/uucp/uucico
tjhorse::1000:Trojan Horse:usr/people/tjhorse:/bin/csh
+mike:nX3,Vu6hB2Q:777:777:Mike Smith:/usr/people/
  mikes:/bin/csh
+izzy::500:500:Izzy Newton:/usr/people/Izzy:bin/csh
```

the YP Master's password map will be consulted whenever users *mike* or *izzy* attempt to login, since the entries in these lines will override the Client password file's entry. When *tjhorse* attempts to login, however, the Client file entry will be used.

Note the following Client password file entries:

```
root::0:0:0000-Admin(0000):/:/bin/csh
daemon:*:1:1:0000-Admin (0000):/:
bin:*:2:2:0000-Admin(0000):/bin:
sys:*:4:0:0000-Admin(0000):usr/src:
+::0:0:::
```

The last line will force the use of the YP Master's password map for *all* user logins except those related to local system accounts.

## the group file

Just as you can control individual user accounts with the Yellow Pages, you can also control group access. Note the entries in the */etc/group* file:

```
sys::0:root,bin,sys,adm
root::0:root
daemon::1:root
bin::2:root,bin,daemon
acctg::990:heibein, stermitz, schmelzer
+proj88::992:most, woo, roy, marx
+proj90::1990:harbaugh, houston, libby
+mktg::1000:,smith, anderson, chin, williams
+oper::30:miller, hiller, diller wickes
```

Again, the group entries preceded by a "+" character will be validated using the YP Master Server's group map.

A Client */etc/group* containing a single entry:

```
+:
```

will force all group checking to be performed using the Master Server's group map.

## network host equivalency

The configuration file */etc/netgroup* is to the network what */etc/groups* is to the workstation. The YP map compiled from the *netgroup* file can be used to control remote logins to a workstation. By making entries into the local */etc/hosts.equiv* file, like so:

```
+@netgroupname
```

or:

```
-@netgroupname
```

you either allow (with the "+" character) or exclude (using the "-") members of the netgroup that have been specified after the @ sign. Note that the netgroup must be a valid entry in the Master Server's netgroup map.

## conclusions

While not a panacea, use of YP services can help to simplify the network administrator's job. By centralizing many of the workstation and network configuration files, the YP service gives the administrator a great deal of control over user and group access to network hosts, and it ensures a large degree of consistency among the workstations using its services through automatic YP map propagation.

*Michael Smith is a systems engineer at SGI responsible for the development of the Network Adminstration course that Silicon Graphics' Customer Education Department offers for the IRIS 4D family.*

# bouquets and broadsides

## let's be perfectly clear

*Dear Editor,*

I would like to correct several errors in your description of your Fall 1988 cover. You are correct when you state that the structures on the cover appear to be designed according to the principles of tensegrity. However, you then imply that tensegrity is a system invented by the artist, and state that the term was coined by reviewers of the artist's work.

In fact, tensegrity was developed many years ago by R. Buckminster Fuller, an American genius whose work spans from automobiles to geodesic domes. The ideas came from Fuller's insight that although traditional buildings are built entirely out of compressive members (*e.g.* bricks laid upon bricks, each experiencing compressive forces), new materials have created the opportunity to build with tensile members of great strength (*e.g.* cables and ropes). One may consider a suspension bridge as a first step toward merging the two, where the posts at either end are compressed against the sea bottom, while the steel cables, under tension, distribute the post's support to the span.

Purely compressive structures cannot adapt to external forces: they either hold or break. A structure built with both compressive and tensile components sits in equilibrium; under stress, it deforms and adapts to the external forces long before it breaks entirely. Fuller invented the word *tensegrity* to describe those structures which successfully merge *tensile* and compressive forces into a whole with structural in*tegrity*.

Perhaps the best book on the subject is *Tensegrity*, by Anthony Pugh (a student of Fuller's), published by the University of California.

*Andrew Glassner*
*Xerox Palo Alto Research Center*
*Palo Alto, CA*

*While it's true that Buckminster Fuller coined the term "tensegrity", what's less well known is that a work by Kenneth Snelson inspired him to do so. Snelson, while studying under Fuller at Black Mountain College, invented the structural form with his 1948 sculpture, "Early X Piece". Fuller, who was much impressed, began speaking and writing about the invention shortly thereafter, describing Snelson's discontinuous compression principle as "tensegrity". To the best of our knowledge, Fuller never claimed to be the inventor, but neither did he deny it. A reasonably concise chronology of the interactions between Fuller and Snelson can be found on pp. 160-161 of Mary Emma Harris' book, "The Arts at Black Mountain College" (MIT Press, 1987).*

*The IRIS Universe invites readers to comment about articles presented in the magazine and to air views on topics of relevance to the IRIS community. Address all correspondence to "Editor, IRIS Universe, 2011 N. Shoreline Boulevard, P.O. Box 7311, Mail Stop 130, Mountain View, CA 94039-7311" or send e-mail to <user@sgi.com>. Letters may be edited for clarity and/or length.*

# product news

Introl Corporation (St. Paul, MN) has introduced a 2300 MB tape backup system, the Sterling 2300T, for SGI workstations. Using *helical scan* technology, long under development in the video tape arena, the system reads and writes to cartridges small enough to fit into a shirt pocket (2-1/2" by 3-1/2") and has a mean time between failure of more than 20,000 hours. The quantity-one price for a complete subsystem is $10,900, and availability is 30 days ARO. Call 612/631-7600 for more information.

ANSYS Revision 4.3a, from Swanson Analysis Systems (Houston, PA) is now available across the entire IRIS 4D product line. A general-purpose, finite-element program for engineering analysis that includes preprocessing, solution, and post-processing, ANSYS is used in a wide variety of disciplines to solve mechanical, thermal, and electronic problems. Call 412/746-9494 for more information.

Silicon Graphics shipped more than 1000 Personal IRIS systems within three months of announcing the machine. The 1000th unit was sent to McDonnell-Douglas. Other volume end-users of the Personal IRIS include Boeing Corporation, Pacific Data Images, Princeton University, and Cray Research.

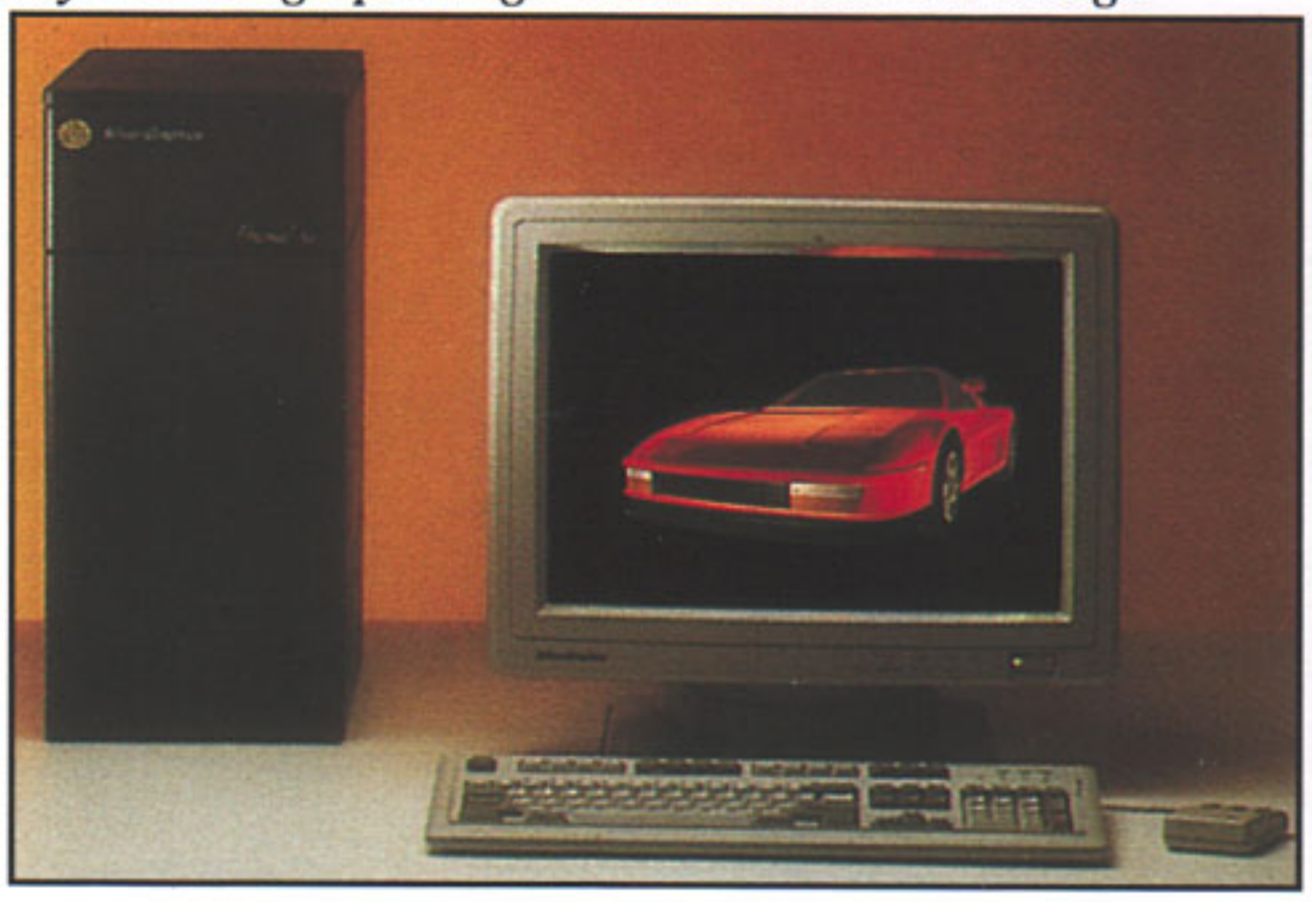
FULL
COLOR
COMPUTING, INC.

(203) 270-1082

39B Mill Plain Road, Suite 299 Danbury, CT 06811

# Full Color Publisher

*...your design package used to create an image*



Sports car image created using the ALIAS/2 design system from Alias Research, Inc.

## It's...

not just sophisticated page layout.
not just a paint system.
not just photo retouch.
not just image manipulation.
not just text handling.

Full Color Publisher
from
Full Color Computing, Inc.,
an innovator in full color publishing
and image processing technologies.

Available now
for all Silicon Graphics platforms
from the 31xx series to the Personal IRIS.

Supports most major product file
formats, scanners and output devices.

*...a scanned image for the background*


*...the final result*

*changed color of car*
*touched up windshield*
*removed lines in road*
*touched up grass*
*added shadow under car*
*increased contrast*
*added text*

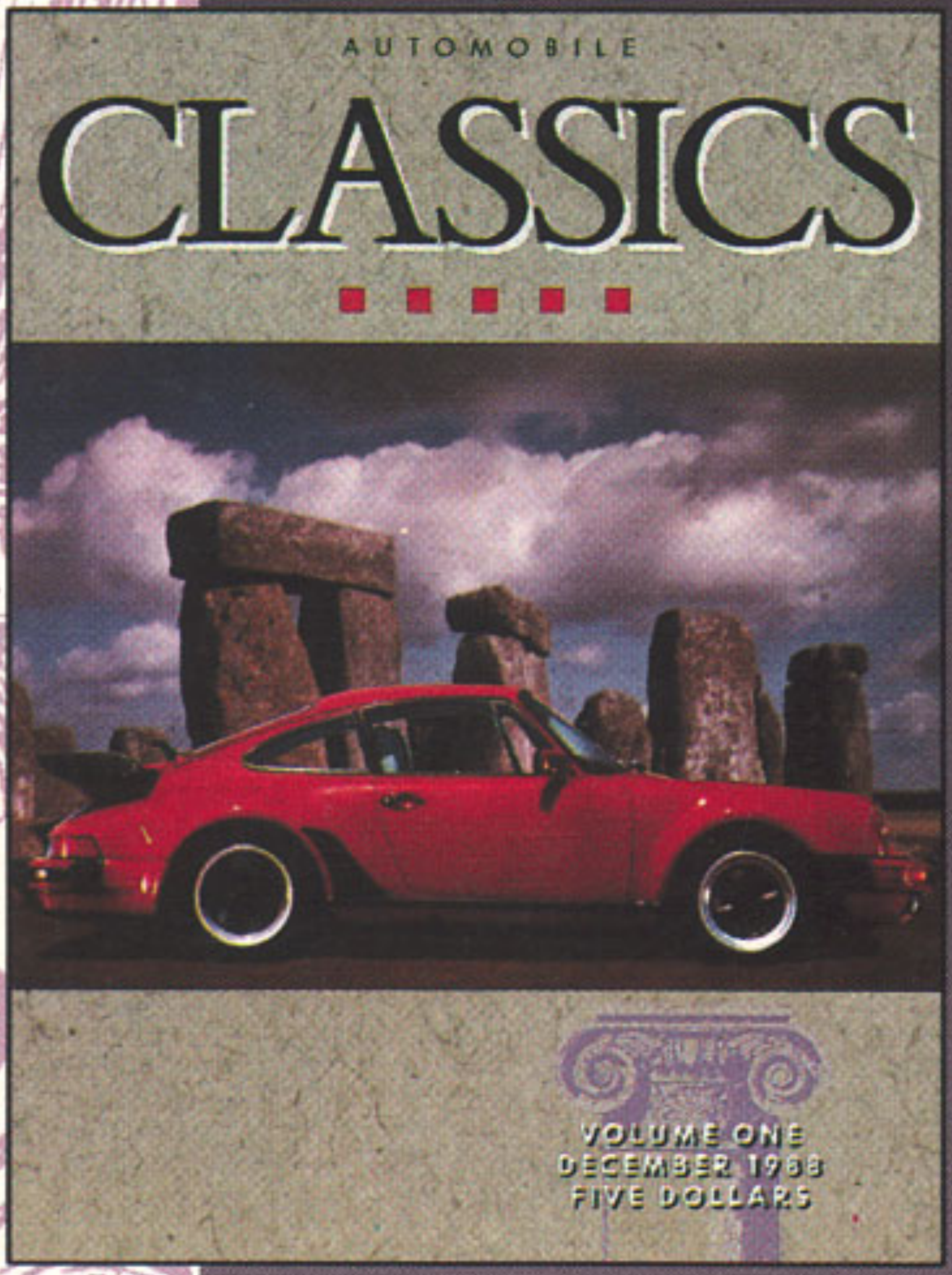
## It's...

just everything you need to produce results like this!

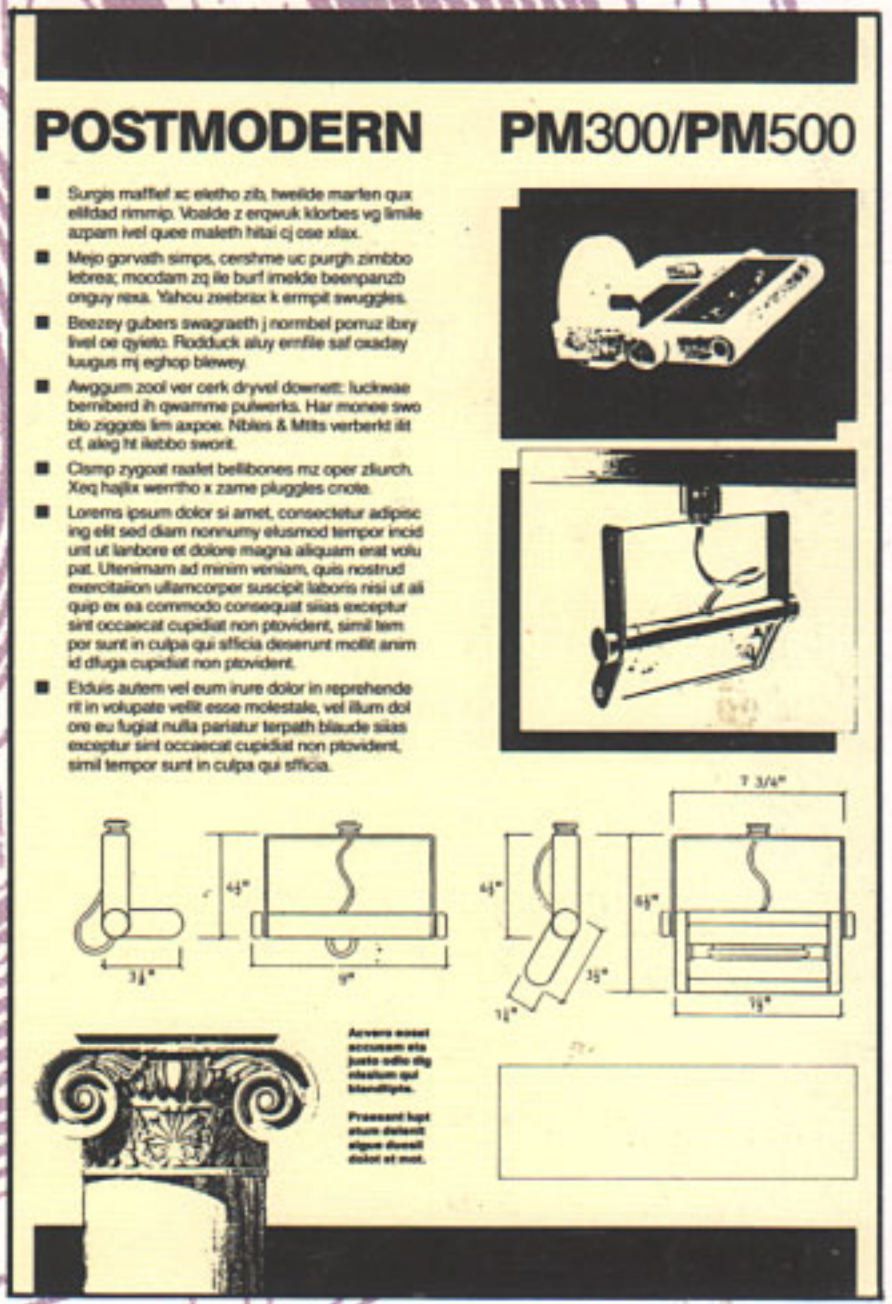
Now available for the Personal Iris 4D/20
with support for:
Wavefront Cubicomp Alias

Today's world of computer aided production and presentation requires a myriad of visual techniques to expound upon the design process. Until now, one of the most important pieces was missing.

Introducing Artisan, a unique software solution for a multitude of painting and illustration needs. Available for use on all Silicon Graphics workstations, Artisan delivers the functionality necessary to add imaging capabilities to a wide range of applications. From animation to CAD, presentations to image processing, nothing covers the spectrum like Artisan.








# Classic Paint Job

Artisan is a trademark of Media Logic Incorporated

a r t i s a n™

For additional information or a demonstration tape, contact:

**Media Logic Incorporated**
**2501 Colorado Avenue, Suite 350**
**Santa Monica, CA 90404**
**(213) 453-7744, (213) 453-9565 Fax**